UNIVERSITÉ DE LIMOGES
ÉCOLE DOCTORALE "Sciences et Ingénierie pour l'Information"
FACULTÉ DES SCIENCES ET TECHNIQUES

Département de Mathématiques et Informatique - Laboratoire XLIM

Thèse N°

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE LIMOGES

Discipline / Spécialité : Informatique

présentée et soutenue par

Boussad AIT-SALEM

le 12/07/2011

Sécurisation des Réseaux Ad hoc :
Systèmes de Confiance et de Détection de Répliques

Thèse dirigée par **Carlos Aguilar Melchor, Philippe Gaborit et Karim Tamine**

JURY

Rapporteurs

M. Christophe BIDAN	Professeur Assistant à Supélec Rennes, HDR de l'Université de Rennes 1
M. Alexis BONNECAZE	Professeur des universités à l'Université de la Méditerranée

Examinateurs

M. Carlos AGUILAR MELCHOR	Maître de conférences à l'Université de Limoges
M. Yves DESWARTE	Directeur de Recherche CNRS au LAAS-CNRS de Toulouse
M. Philippe GABORIT	Professeur des universités à l'Université de Limoges
M. Karim TAMINE	Maître de conférences à l'Université de Limoges

À mes parents…

Remerciements

Je tiens tout d'abord à remercier mes encadrants : Carlos Aguilar Melchor, Philippe Gaborit et Karim Tamine pour la confiance qu'ils m'ont accordée, ainsi que pour toute l'aide et tous les encouragements qu'ils m'ont apportés dans les moments les plus difficiles.

J'adresse mes sincères remerciements à M. Christophe Bidan et à M. Alexis Bonnecaze pour avoir accepté la charge de rapporteur malgré leur emploi du temps chargé, ainsi qu'à M. Yves Deswarte pour son rôle d'examinateur.

Merci à tous les doctorants et tout le personnel du département de mathématiques et informatique. Je pense tout particulièrement à Agnès, Ahmadou, Alex, Amaury, Amine, Cyril, Emmanuelle, Julien, Mickael, Nadir, Nassima, Richard et Slim.

Ma pensée va également vers ma femme qui m'a toujours soutenu dans les moments difficiles et qui a toujours su trouver les moyens pour me réconforter.

Enfin, je terminerai ces remerciements par une pensée particulière pour mes parents, ma sœur Lynda et mes deux frères Aghiles et Juba qui m'ont constamment soutenu dans mes projets d'études.

Résumé

L'évolution rapide des technologies sans fil a permis l'émergence de nouveaux types de réseaux très dynamiques avec des architectures fortement décentralisées et dont les services sont organisés de manière autonome. Ces spécificités sont à double tranchant, car, d'une part, elles permettent une mise en place peu coûteuse et rapide de ces réseaux, et d'autre part, elles engendrent des difficultés lors de la conception de certains services tels que le routage, la qualité de service, la sécurité, etc.

Dans le cadre de cette thèse, on s'est intéressé à la sécurité des systèmes de confiance dans ces réseaux.

Les systèmes de confiance et de réputation ont été introduits en informatique dans le but d'évaluer la qualité des ressources et la fiabilité des entités vis-à-vis d'un service donné. L'utilisation des systèmes de confiance dans des environnements avec infrastructure repose généralement sur une entité de confiance centrale qui réalise les principales tâches : authentification des utilisateurs, calcul et agrégation des valeurs de confiance, protection des données privées, répudiation des utilisateurs malveillants, etc. Cependant, les réseaux sans infrastructure tels que les réseaux ad hoc perdent une grande partie de leur intérêt s'ils se basent sur des systèmes centralisés pour leurs services. Ainsi, on essaye d'obtenir des protocoles n'utilisant pas une entité centrale.

En l'absence d'une entité centrale, les entités doivent s'évaluer les unes les autres et agréger les résultats. Pour que les évaluations soient faites sans crainte, il faut que les valeurs de celles-ci restent secrètes jusqu'à l'agrégation. Dans ce manuscrit, nous allons essayer d'apporter une solution à ce problème en proposant un schéma, basé sur un chiffrement homomorphe [RAD78] et sur le protocole d'envoi superposé [Cha88], qui permette à chaque entité de calculer la valeur de confiance d'une autre entité tout en protégeant les données privées des entités engagées dans ce calcul. Nous avons construit notre schéma dans le cadre du modèle semi-honnête défini par Goldreich dans [Gol04]. Nous donnons, à ce titre, des preuves de sécurité qui montrent que notre schéma respecte ce modèle et en particulier

qu'il est résistant à une collusion de $n-1$ participants, n étant le nombre de participants au calcul. En termes de performance, il est également démontré que notre proposition réduit le coût des communications par rapport aux propositions antérieures. Les résultats que nous présentons dans ce domaine ont donné lieu à une publication dans la conférence "IEEE Network Computing and Applications" [AAG09].

Par ailleurs, l'absence d'une entité de confiance (p. ex., une autorité de certification) peut aussi causer des problèmes au niveau de la gestion et de l'authentification des utilisateurs. Un attaquant peut manipuler la plage des identités à sa guise, à savoir : créer de nouvelles identités, usurper celles existantes, etc. Des méthodes d'authentification adaptées aux réseaux ad hoc ont été proposées. Cependant, dans les réseaux ad hoc tels que les réseaux de capteurs, un autre problème apparaît. En effet, à cause de la faible protection physique des capteurs et de l'environnement hostile dans lequel ils sont déployés, un attaquant peut facilement construire plusieurs copies d'un nœud capturé et ainsi outrepasser le mécanisme d'authentification mis en place. Ceci peut en particulier porter atteinte aux systèmes de confiance étudiés dans ce manuscrit quand on se place dans le contexte des ces réseaux de capteurs.

Parno et *al.* sont les premiers à proposer dans [PPG05] un protocole de détection de nœuds répliqués basé sur la localisation géographique des capteurs, suivant une approche de détection passive et distribuée. Le principe de base de cette méthode consiste à imposer à chaque nœud du réseau la déclaration de ses coordonnées géographiques à un ensemble de nœuds du réseau appelés témoins. Chaque témoin a pour rôle de vérifier la validité des déclarations reçues (positions géographiques cohérentes, signatures, etc.) et de révoquer les nœuds concernés par d'éventuels conflits. Si la méthode de sélection de témoins dévoile des informations sur l'identité ou la localisation géographique des futurs témoins, un attaquant peut exploiter ces informations afin de les neutraliser et permettre ainsi à ses répliques d'être indétectables. Appliqué à un réseau de n capteurs, le protocole de Parno et *al.* engendre $O(\sqrt{n})$ messages par nœud, alors que les approches traditionnelles engendrent $O(n)$ messages par nœud. Ce protocole est considéré comme étant le premier protocole non centralisé à offrir des performances intéressantes. Malheureusement, chaque capteur doit sauvegarder $O(\sqrt{n})$ coordonnées géographiques signées. Ceci est un défaut important sachant que la mémoire des capteurs est une ressource très limitée. Par ailleurs, d'après des simulations réalisées dans [CDPMM07], il a été démontré que le nombre de témoins n'est pas uniformément réparti sur toute la surface du réseau, ce qui influe négativement sur la résistance du protocole aux attaques et sur la répartition des coûts dans le réseau.

D'autres protocoles améliorant la sécurité et les performances du protocole de Parno

et *al.* ont été proposés. L'un des travaux les plus intéressants est celui proposé dans [CDPMM07]. En plus d'avoir une complexité constante en termes d'occupation mémoire, sa méthode de sélection de témoins ne divulgue aucune information sur les futurs témoins. Malheureusement, ce protocole nécessite un coût de communication supplémentaire pour mettre en place, à chaque round de détection, un nombre aléatoire dont la génération est réalisée par une tierce partie de confiance. Afin de réduire les coûts considérables qui découleraient d'une diffusion de ce nombre aléatoire dans tout le réseau, sa distribution est déléguée soit aux stations de base, soit à un protocole additionnel d'élection de leaders, réputé pour être très sensible à une attaque par déni de service.

Dans ce manuscrit, nous proposons une nouvelle approche de détection qui est autonome, résistante aux attaques, engendrant un coût de communication en $O(\sqrt{n})$ et ne nécessitant aucune sauvegarde au niveau des capteurs. Ces résultats ont donné lieu à une publication dans le journal "International Journal of Computer Science and Network Security" [AAGT09].

Nous présentons également en annexe nos travaux sur le routage dans les réseaux ad hoc. Cette tâche, qui est réalisée par des nœuds spéciaux (routeurs) dans les réseaux classiques, est assurée en commun par tous les nœuds d'un réseau ad hoc. En effet, chaque nœud d'un réseau a pour mission d'aider les autres dans la recherche de routes et de faire suivre les messages de ses voisins. Eu égard aux ressources matérielles limitées, le routage est délicat du fait qu'il faut trouver des solutions optimales pouvant allier performance et efficacité. A cet effet, nous proposons un protocole de routage hybride basé sur les systèmes multiagents. La proposition et l'étude de ce protocole ont donné lieu à une publication dans la conférence "International Conference on Availability, Reliability and Security" [AAGT08].

Abstract

The rapid advances in wireless technologies have allowed the emergence of new types of dynamic networks with highly decentralized architectures in which services are organized in an autonomous manner. These specific features are a double-edged sword, because, on the one hand, they allow a rapid and cheap establishment of these networks, and on the other hand, they give rise to additional difficulties in the design of services such as routing, QoS, security, etc.

In the context of this thesis, we have focused on the security of trust systems in these networks.

The trust systems have been introduced in computer science in order to evaluate the resource quality and the reliability of entities to a given service. The use of trust systems in environments with an infrastructure usually relies on a central trusted entity that performs the main tasks : user authentication, calculation and aggregation of trust values, protection of private data, repudiation of malicious users, etc. However, the networks without infrastructure such as ad hoc networks lose a large part of their interest if they rely on centralized systems in order to implement their services. Hence, we try to propose protocols that do not rely on any central entity.

In the absence of a central entity, the entities must evaluate each others and aggregate their results. To make an assessment without fear, it is necessary that their values remain secret until the aggregation. In this manuscript, we try to solve this problem by proposing a new scheme based on an homomorphic encryption [RAD78] and on the superposed sending protocol [Cha88]. This scheme enables each entity to compute the trust value of another entity while, at the same time, protecting the private data involved in this computation. We built our scheme in a semi-honest model defined by Goldreich in [Golo4]. We give, for this purpose, security proofs demonstrating that our scheme respects this model and in particular that is resistant to collusion of up to $n-1$ participants, where n is the number of participants in the system. In terms of performance, it is also proved that our proposal

reduces communication costs compared to previous proposals. The results that we present in this area have led to a publication in the conference "IEEE Network Computing and Applications" [AAG09].

Moreover, the absence of a trusted entity (e.g., a certification authority) can also cause problems in terms of users authentication. An attacker could easily manipulate the range of identities, namely : create new identities, impersonate existing identities, etc. Authentication methods suitable for ad hoc networks have been proposed. However, in ad hoc networks such as sensor networks, another problem appears. Indeed, due to the weaknesses in the physical protection of sensors and to the hostile environment in which they are deployed, an attacker can easily build multiple copies of a captured node and thus bypass the authentication mechanism in place. This can especially affect the trust systems studied in this manuscript when we put ourselves in the context of these sensor networks.

Parno et *al.* are the first that proposed in [PPG05] a distributed protocol for detecting a node replication attack based on the geographical location of sensors. The basic idea of this approach is to force each node to send its signed location claim to a set of nodes called witnesses. The role of each witness is to check the received location claims and to revoke the nodes involved in conflicts. However, if the witnesses selection process reveals information about the identity or the location of future witnesses, an attacker can exploit this information in order to neutralize the witnesses and allow its replicas to be undetectable. Applied to a network of n sensors, their protocol generates $O(\sqrt{n})$ messages per node, while traditional approaches generate $O(n)$ messages per node. The protocol of Parno et *al.* is considered as the first non-centralized protocol that provides interesting performances. Unfortunately, each mote needs to save $O(\sqrt{n})$ signed location claims. This fact constitutes a major obstacle to its implementation in practice, since the memory of sensors is a very limited resource. Furthermore, according to simulations performed in [CDPMM07], it was shown that the number of witnesses is not uniformly distributed over the network. This fact adversely affects the resilient of the protocol against attacks and the load balancing in the network.

Other protocols improving the security and the performances of Parno et *al.* protocol have been proposed. One of the most interesting works has been proposed in [CDPMM07]. Besides having a constant memory complexity, the used selection method does not disclose any information about future witnesses. Unfortunately, this protocol requires an extra communication cost in order to set up at each detection round a random number that is generated by a trusted party. In order to reduce the significant costs that would result from broadcasting the random number across the network, its distribution is delegated either to base stations or to an additional leader election protocol which is reputed to be very sensitive

to denial of service attacks.

In this manuscript, we propose a new detection approach which is autonomous, resilient to attacks, generates $O(\sqrt{n})$ communication complexity and does not require any backup on sensors. This proposition resulted in a publication in the journal "International Journal of Computer Science and Network Security" [AAGT09].

We also present in appendix A our works relating to routing protocols for ad hoc networks. The routing task, which is performed by special nodes (routers) in traditional networks, is ensured jointly by all nodes in ad hoc networks. Indeed, each node of an ad hoc network must help other nodes in finding routes and forward the messages of its neighbors. In view of limited hardware resources, the routing is more difficult because we must find optimal solutions that could combine performance and efficiency. To this end, we propose an hybrid routing protocol based on multiagent systems. The proposal and the study of this protocol gave rise to a publication in the conference "International Conference on Availability, Reliability and Security" [AAGT08].

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction générale

Un réseau sans fil ad hoc est constitué d'un ensemble de nœuds (terminaux, ordinateurs portables, PDAs, smartphones, capteurs, etc.) pouvant être mobiles. La mobilité d'un nœud peut se limiter au déploiement du réseau, comme c'est le cas des réseaux de capteurs, ou bien persister durant tout le cycle de vie du réseau, comme c'est le cas des réseaux mobiles ad hoc (*Mobile Ad hoc Networks ; MANETs*). Les nœuds sont équipés d'une interface de communication sans fil (radio) à faible portée et peuvent communiquer (envoyer ou recevoir des données) avec des nœuds qui sont à l'intérieur de leur portée radio (c.-à-d. avec leurs nœuds voisins immédiats).

Trois spécificités importantes de ce type de réseaux rendent la conception de leurs différentes fonctionnalités délicate. La première spécificité est que les interconnexions des nœuds sont instables à cause d'une part, de l'utilisation de la technologie sans fil et d'autre part, de la mobilité fréquente des nœuds. La seconde spécificité réside dans le fait que les nœuds ont une capacité de calcul, de communication, d'énergie et de stockage très limitée, et ce, particulièrement dans les réseaux de capteurs. La troisième spécificité est qu'ils doivent être capables de former un réseau spontanément sans avoir besoin d'une infrastructure fixe spécifique. Afin de surmonter cet obstacle, les nœuds d'un réseau ad hoc s'auto-organisent et coopèrent afin d'assurer toutes les opérations indispensables à un réseau de communication : routage de données ; configuration ; sécurisation, etc.

Toutes ces spécificités imposent des contraintes supplémentaires aux concepteurs et aux développeurs de services informatiques dédiés aux réseaux ad hoc et les poussent à réfléchir à de nouvelles approches. Dans cette thèse, on s'est focalisé sur deux problématiques :

Problématique 1

Au-delà des approches traditionnelles utilisant les techniques cryptographiques classiques pour assurer les propriétés de sécurité les plus importantes (confidentialité, intégrité, authentification et non-répudiation), des mécanismes d'établissement de la confiance et de la réputation ont été proposés afin de créer et de gérer la confiance entre les entités actives d'un système informatique [JIB07]. En effet, les systèmes de confiance et de réputation permettent, à chaque entité participant à un protocole donné, de mesurer la fiabilité d'une autre entité avant de décider d'interagir ou d'entrer en communication avec elle. Ils représentent donc un moyen d'inciter ces entités à un bon comportement et à toujours offrir des services ou des ressources de qualité. Dans le cas des systèmes de réputation, l'idée de base est de permettre aux entités de s'évaluer entre elles, par exemple à la fin de chaque transaction, puis d'utiliser des agrégats de ces évaluations (recommandations) afin de déduire la note de réputation d'une entité donnée. Dans le cas des systèmes de confiance, l'idée de base est d'analyser et de combiner des réseaux et des chemins de relations de confiance afin de déduire des mesures de confiance subjectives permettant d'évaluer la fiabilité d'une entité donnée. Les systèmes de confiance et de réputation ont de nombreuses applications dans les environnements distribués tels que les réseaux ad hoc [EGB02, TB04, SDB03], les réseaux de capteurs, les réseaux pair-à-pair [THVW05] et bien sûr dans le cadre des réseaux classiques (par exemple pour les applications de commerce électronique telles que les ventes aux enchères).

Le fonctionnement d'un système de confiance et de réputation centralisé peut s'appuyer sur une entité de confiance qui s'occupe des principales tâches : authentification des utilisateurs, collecte des évaluations, agrégation et combinaison des évaluations, protection des données privées des entités participantes, sanction ou exclusion des utilisateurs malveillants. L'obtention d'un système de confiance et de réputation dans un environnement sans infrastructure tel qu'un réseau ad hoc est un peu plus délicate, vu qu'on ne peut généralement pas désigner une entité de confiance reconnue par tous les participants.

En l'absence d'une autorité centrale, on peut concevoir un système où chaque entité évalue la fiabilité des autres entités ou la qualité des services offerts, sauvegarde localement ces évaluations et contribue au calcul des notes de confiance des autres participants en envoyant ses évaluations personnelles sous forme de recommandations publiques. Toutefois, ces recommandations représentent les opinions personnelles des entités et renferment parfois des informations significatives sur leurs intentions, elles doivent de ce fait être considérées comme étant des informations sensibles nécessitant une protection. Nous

voulons pour preuve les deux exemples suivants. Le premier exemple est extrait d'une étude empirique réalisée dans [RZ02] sur un ensemble de données récoltées sur le site de ventes aux enchères eBay. Cette étude montre clairement qu'il existe une forte corrélation entre les évaluations des acheteurs et celles des vendeurs et que plus de 99% des évaluations effectuées par les acheteurs sont positives. Ces résultats peuvent être expliqués par le fait que lorsque les identités (pseudo-identités) des évaluateurs sont connues, leurs évaluations sont réalisées de manière stratégique pour des raisons de réciprocité ou de peur d'une éventuelle vengeance de celui qui est évalué. Donc, ces évaluations ne reflètent guère la fiabilité du vendeur. Il est, par conséquent, important de préserver la confidentialité de ces évaluations. Le deuxième exemple concerne l'utilisation d'un système de réputation afin de surveiller la fonction de routage dans un réseau ad hoc. Dans un tel mécanisme, les nœuds s'appuient sur les notes de confiance de leurs voisins afin de sélectionner la route la plus sûre en choisissant ceux dont la note de confiance est la plus élevée. Dans le cas où le processus de calcul de ces notes de confiance n'offre aucune protection aux recommandations échangées, un attaquant peut profiter de cette fuite d'informations pour deviner la route qui sera choisie par un nœud et ainsi lui permettre de mieux cibler son attaque.

Dans un système de confiance et de réputation, différentes informations peuvent être considérées comme privées : les résultats des différents calculs de confiance, les recommandations, le niveau de confiance accordé à ces recommandations (poids affectés aux recommandations) et les identités (pseudo-identités) des participants. Dans cette thèse, nous allons nous intéresser essentiellement à la protection des trois premières catégories d'informations citées sachant que la préservation des identités est traitée dans de nombreux travaux autour de l'anonymat dans les réseaux de communication.

Le problème de protection de ces données privées peut être résolu de deux manières différentes. La première consiste à masquer le lien entre la pseudo-identité de celui qui recommande et la recommandation. Cette approche peut être intéressante dans des systèmes de réputation basiques qui ne tiennent pas compte des poids de ces recommandations, mais dans d'autres cas elle s'avère inappropriée ou complique davantage le modèle de calcul utilisé. De plus, ce mécanisme complique la tâche des systèmes de gestion de confiance qui s'appuient justement sur ces identités pour identifier les utilisateurs malveillants.

La seconde, qui fait l'objet du chapitre 3, consiste à permettre à chaque entité de faire ses calculs sur la base des recommandations des uns et des autres ainsi que sur la base de la confiance qu'elle accorde à celles-ci tout en préservant leur confidentialité. En d'autres termes, nous voulons répondre à la question suivante :

Comment réaliser le calcul des notes confiance et de réputation de manière distribuée et coopérative sans qu'aucune partie ne puisse connaître les informations privées des autres ?

Nous avons pu répondre à cette question en proposant un nouveau protocole inspiré de ce qui se fait dans le domaine des calculs distribués sécuritaires multiparties (Secure Multi Party Computation, SMPC) introduits par A. C. Yao [Yao82]. Les protocoles utilisés dans le SMPC s'appuient sur des primitives cryptographiques permettant à différentes parties de réaliser des calculs, basés sur leurs données privées $(p_1, p_2, ..., p_n)$, de manière à ce que tout le monde connaisse le résultat final du calcul : $f(p_1, p_2, ..., p_n)$, mais aucune partie ne puisse déduire les données privées des autres. En plus de nombreux schémas génériques proposés dans le domaine du SMPC, il existe des applications pratiques dans de nombreuses disciplines de l'informatique. Par exemple, dans les bases de données, les calculs scientifiques, la fouille de données, l'intégration de données, l'agrégation de données dans les réseaux de capteurs, etc. Le protocole que nous proposons au chapitre 3 est basé sur deux primitives cryptographiques de base : le chiffrement homomorphe [RAD78] et l'envoi superposé [Cha88]. Grâce à l'utilisation de ces deux primitives nous améliorons une première solution donnée par Yao et *al.* dans [YTP07]. Nous avons construit notre schéma dans un modèle semi-honnête défini par Goldreich dans [Golo4]. Nous donnons, à ce titre, des preuves de sécurité qui montrent que notre schéma respecte ce modèle et en particulier qu'il est résistant à une collusion constituée de $\tau \in [1, n-1]$ participants, n étant le nombre de participants au calcul. En termes de performance, il est également démontré que notre proposition réduit le coût des communications de la proposition de Yao et *al.*. Ces résultats ont donné lieu à une publication dans la conférence "IEEE Network Computing and Applications" [AAGo9].

Problématique 2

Les systèmes de réputation et de confiance dépendent fortement de la possibilité de lier une identité à une entité de façon pérenne. Dans les réseaux ad hoc, l'absence d'une entité de confiance (p. ex., une autorité de certification) rend difficile la mise en place d'un système d'authentification permettant de gérer efficacement les identités des utilisateurs. Dans le cas où le réseau est dépourvu d'un tel mécanisme, un attaquant peut aisément manipuler la plage des identités en créant par exemple de nouvelles identités ou en usurpant celles existantes.

Ceci peut perturber le fonctionnement des systèmes de confiance. Par exemple, un

attaquant peut créer un nombre disproportionné de nœuds qui vont se comporter à tour de rôle de manière malveillante et se surévaluer entre eux afin de rester le plus longtemps possible dans le réseau, sans se faire repérer par le système de réputation. Un attaquant peut aussi créer une collusion importante de nœuds afin de baisser la réputation d'un nœud honnête du réseau.

Des méthodes d'authentification adaptées aux réseaux ad hoc ont été proposées. Cependant, dans les réseaux de capteurs, ces solutions sont peu sécurisantes. En effet, à cause de la faible protection physique des capteurs et de l'environnement hostile dans lequel ils sont déployés, un attaquant peut facilement construire plusieurs copies d'un nœud capturé et ainsi contourner le mécanisme d'authentification mis en place. Peu de solutions ont été proposées pour remédier à ce problème et la majorité d'entre elles ne sont pas applicables en pratique aux réseaux de capteurs, principalement pour des raisons de coût.

La limitation des ressources physiques des réseaux de capteurs est beaucoup plus importante que dans les réseaux ad hoc classiques, ce qui impose de sérieux défis lors de la conception des protocoles dédiés à ce type de réseaux. Les capteurs ont une capacité de calcul, de communication et de stockage très limitée. À titre d'exemple, la mémoire RAM varie entre 4 KB pour MICA 2 et 10 KB pour Telos B mote.

De nombreuses mesures de sécurité ont été proposées afin de remédier à la réplication. Nous pouvons les classifier selon qu'elles soient centralisées ou distribuées et selon la manière de sélectionner les nœuds témoins.

Dans une approche centralisée, chaque nœud du réseau envoie la liste des déclarations de position de ses voisins directs à la station de base qui les examine afin de détecter d'éventuels conflits. Ceci engendre une congestion très forte des routes vers la station de base. Le défaut majeur en termes de sécurité de cette approche est qu'elle repose sur un seul point de vulnérabilité.

Parno et *al.* sont les premiers à proposer dans [PPG05] des protocoles de détection distribués suivant une approche passive. Leur meilleure proposition engendre $O(\sqrt{n})$ messages par nœud. Par ailleurs, chaque capteur doit sauvegarder $O(\sqrt{n})$ coordonnées géographiques signées. Ceci constitue un défaut important sachant que la mémoire des capteurs est une ressource très limitée. En termes de résistance aux attaques, il a été démontré que le nombre de témoins n'est pas uniformément réparti sur toute la surface du réseau (les témoins se trouvent majoritairement au centre du réseau), ce qui influe négativement sur la résistance du protocole aux attaques et sur la répartition des coûts dans le réseau.

D'autres protocoles améliorant la sécurité et les performances du protocole de Parno

et *al.* ont été proposés. L'un des travaux les plus intéressants est celui proposé dans [CDPMM07]. En plus d'avoir une complexité constante en termes d'occupation mémoire, sa méthode de sélection de témoins ne divulgue aucune information sur les futurs témoins. Malheureusement, ce protocole nécessite un coût de communication supplémentaire pour mettre en place, à chaque round de détection, un nombre aléatoire dont la génération est réalisée par une Tierce Partie de Confiance (TPC). Afin de réduire les coûts considérables qui découleraient d'une diffusion de ce nombre aléatoire dans tout le réseau, sa distribution est déléguée soit aux stations de base, soit à un protocole additionnel d'élection de leaders, réputé pour être très sensible aux attaques par déni de service.

Un autre protocole engendrant les mêmes coûts que ce dernier a également été proposé dans [ZAS+07]. Cependant, la méthode de sélection de témoins utilisée est moins résistante aux attaques.

Dans ce manuscrit, nous proposons un nouveau protocole de détection distribué et suivant une approche active de détection. En d'autres termes, c'est les nœuds eux-mêmes qui initient les vérifications. L'idée est de permettre à chaque nœud de vérifier, périodiquement, la non-réplication d'un certain nombre de nœuds choisis au hasard.

Pour un taux de détection avoisinant les 100%, notre solution permet une complexité équivalente aux travaux parus dans [CDPMM07]. Notre approche suppose que parmi les nœuds les plus proches d'une réplique, certains routent les messages vers cette réplique, mais nous n'avons recours ni aux services d'une entité tierce, ni à un protocole additionnel, sujet à des attaques de déni de service. De plus, notre protocole ne divulgue aucune information permettant à un attaquant de prédire les identités ou l'emplacement géographique des futurs témoins. Ces résultats ont donné lieu à une publication dans le journal "International Journal of Computer Science and Network Security" [AAGT09].

Chapitre 2

Les réseaux ad hoc

La prolifération des technologies sans fil n'a cessé de croître ces dernières années. Elles répondent de plus en plus efficacement à nos besoins et exigences en termes de communication. Elles sont devenues incontournables pour assurer le fonctionnement de nombreux services informatiques et des télécommunications tels que la téléphonie, la messagerie électronique et l'accès à Internet.

Il existe de nombreuses raisons à la réussite des technologies sans fil. D'abord, parce que le prix des dispositifs sans fil a considérablement diminué, ce qui a permis aux fournisseurs de services sans-fil de diminuer, en conséquence, les prix de leurs offres de service. De plus, l'installation des réseaux sans fil est moins coûteuse en termes de temps d'installation et de prix en comparaison à une installation filaire. Et enfin, les dispositifs sans fil sont devenus beaucoup plus fiables et leurs performances s'améliorent de plus en plus, rendant ainsi possible l'exploitation de nouveaux services réputés pour être gourmands en bande passante.

Dans les réseaux sans fil, les terminaux transmettent leurs informations par propagation électromagnétique. Le signal émis par un terminal ne peut être intercepté que par les nœuds qui sont suffisamment proches du nœud transmetteur. La distance limite est appelée portée radio d'un nœud (elle est représentée sur la figure 2.1 sous forme d'une zone circulaire autour d'un terminal). Elle ne dépend pas seulement de la qualité du signal de transmission, mais aussi de l'environnement du réseau, par exemple le terrain, les obstacles, etc.

Généralement, on peut trouver plusieurs terminaux sur une petite zone géographique qui veulent transmettre des données sur le support de transmission sans fil. La présence de plusieurs transmissions entamées en même temps à l'intérieur de la portée radio d'un

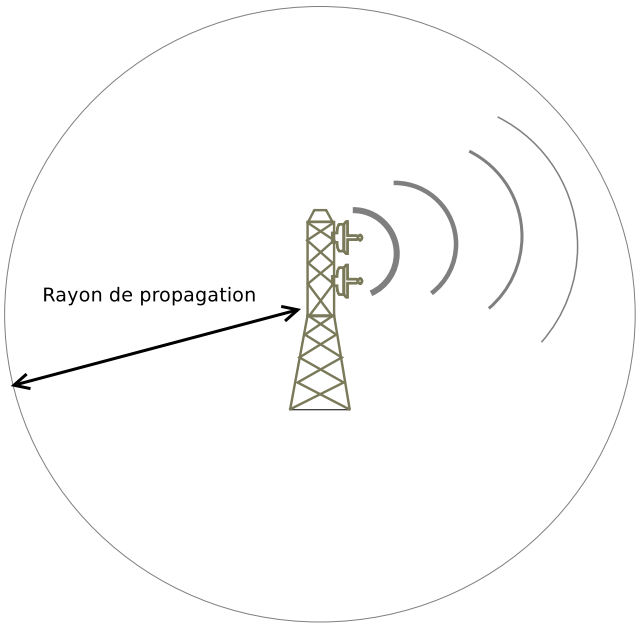

FIGURE 2.1 – Portée radio (zone de couverture) d'un nœud

terminal, peut provoquer des collisions entre elles. On peut comparer ce phénomène à plusieurs individus parlant en même temps à une même personne. Cette dernière sera dans l'incapacité de comprendre l'un ou l'autre de ses interlocuteurs. Afin de remédier à ce problème, il est nécessaire de mettre en place une politique d'accès au support de transmission qui permette d'éviter les collisions ou dans le cas échéant de les minimiser.

Il existe plusieurs schémas [KRD06] permettant le partage du support de transmission sans fil entre plusieurs terminaux. Ils servent à gérer l'accès concurrent au support de transmission et ainsi éviter les collisions. Ces protocoles interviennent au niveau de la couche de liaison de données (la couche MAC). Nous pouvons citer, ici, les plus importants :

- FDMA (Frequency Division Multiple Access) : chaque nœud utilise un domaine de fréquences différent pour communiquer à tout moment ;
- TDMA (Time Division Multiple Access) : tous les nœuds utilisent la même bande de fréquences, mais pas au même moment. L'inconvénient d'une telle approche est qu'elle nécessite une synchronisation très fine ;
- CDMA (Code Division Multiple Access) : tous les nœuds transmettent en même temps en utilisant un codage différent des autres ;

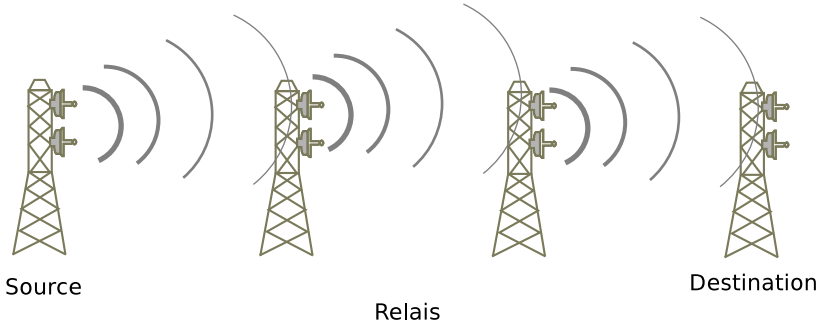

FIGURE 2.2 – Utilisation de relais

- CSMA (Carrier Sense Multiple Access) : quand un nœud veut expédier des données, il commence par écouter le support de transmission et transmet ses trames dès que le canal est disponible.

Il arrive aussi qu'un nœud veuille communiquer avec un autre nœud hors de son champ radio. Pour ce faire, il sollicite les services des nœuds qui sont dans son voisinage pour acheminer son message (faire suivre son message) jusqu' à destination (voir figure 2.2).

On peut citer deux exemples de réseaux sans fil très utilisés dans notre vie quotidienne.

L'exemple le plus connu est sans doute celui des réseaux cellulaires, communément appelés les réseaux GSM (Global System for Mobile Communications), la figure 2.3 montre un exemple type d'une telle architecture. Un réseau cellulaire est constitué de plusieurs *stations de base* fixes à transmission radio réparties sur un très grand territoire. Elles ont pour tâche la réception des données transmises par d'autres bases et leur retransmission vers leurs destinataires. Les utilisateurs finaux sont de petites entités mobiles (p. ex. les téléphones portables) qui communiquent exclusivement avec les stations de base les plus proches. L'interconnexion entre les stations de base est généralement préétablie dès le déploiement du réseau.

Un deuxième exemple de réseaux sans fil, tout aussi répandu, utilise une architecture adaptée aux réseaux locaux sans fil aussi appelés réseaux Wi-Fi (Wireless Fidelity). Ces réseaux sont maintenant largement déployés dans de nombreuses sociétés, universités, aéroports, etc.. Ils fonctionnent sur le même principe de relais que les réseaux cellulaires (stations de base fixes appelées bornes Wi-Fi), sauf que dans le cas du Wi-Fi, les relais sont généralement reliés entre eux par un réseau filaire.

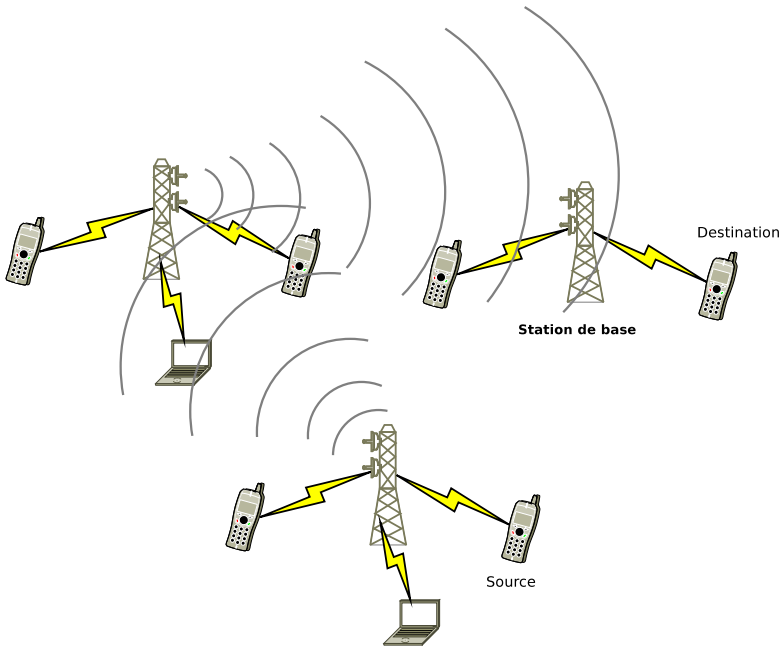

FIGURE 2.3 – Architecture d'un réseau cellulaire

Les types de réseau que nous venons de présenter ci-dessus, reposent sur une infrastructure fixe composée soit de stations de base fixes dans le cas des réseaux cellulaires, soit sur des points d'accès (bornes Wi-Fi) dans le cas des réseaux Wi-Fi. Ce type d'architectures offre une grande puissance et stabilité, mais pour certaines applications, le déploiement de ce type d'infrastructures lourdes est irréalisable faute de temps ou de moyens.

2.1 *Le concept de réseau ad hoc*

Les réseaux mobiles ad hoc, aussi appelés *MANET* (pour *Mobile Ad hoc Network*), sont des réseaux radio sans infrastructure prédéfinie, aptes à se créer et à s'organiser dynamiquement quand plusieurs équipements se trouvent à portée radio les uns des autres. Une des principales caractéristiques de ces réseaux est que les nœuds sont mobiles, et qu'ils peuvent rejoindre ou quitter le réseau de façon dynamique. La figure 2.4 montre une topologie classique d'un réseau ad hoc. Comme le montre cette figure, les nœuds ne s'appuient pas sur un point d'accès ou une station de base pour communiquer entre eux. En fait, les nœuds utilisent leurs voisins immédiats afin de faire suivre leurs communications. Les nœuds

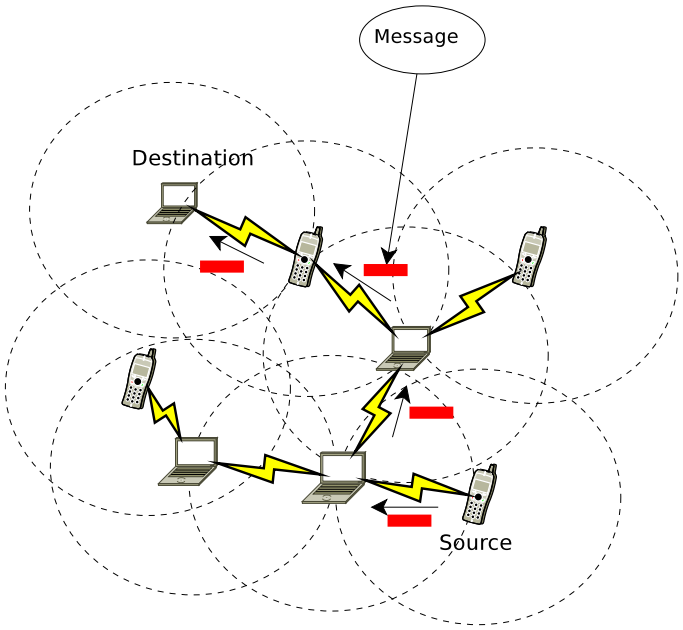

FIGURE 2.4 – Architecture d'un réseau ad hoc

commencent par découvrir leur voisinage (les nœuds qui sont à l'intérieur de leur zone de couverture) et à chaque fois que l'un d'entre eux souhaite communiquer, il envoie son message à ses voisins qui à leur tour l'envoient à leurs voisins et ainsi de suite. Ce processus est répété jusqu'à ce que le message atteigne le nœud destinataire. Ainsi, il est possible de créer de manière spontanée des réseaux à partir d'un ensemble de nœuds suffisamment proches les uns des autres.

Un des paradigmes des réseaux ad hoc est celui de la mise en place rapide d'un réseau de communication dans le cas d'une catastrophe sur des zones dépourvues d'infrastructures ou encore dans le cas où l'infrastructure existante est hors service, voire complètement détruite. Les exemples les plus réalistes sont les catastrophes naturelles comme les tremblements de terre, les inondations, etc. Les équipes de secours ont besoin de mettre rapidement en place un réseau de communication pour coordonner les recherches, les informations logistiques, voire désenclaver les populations isolées en leur offrant rapidement un moyen de communication.

Les nœuds des réseaux ad hoc peuvent être de natures très diverses (ordinateur

portable, PDA, caméra sans fil, capteur, téléphone mobile, lecteur MP3, console de jeux, etc.) et avoir des caractéristiques matérielles et logicielles variées (débit de transmission, mémoire, batterie, unité centrale, etc.). Ces nœuds peuvent communiquer en utilisant une ou plusieurs technologies sans fil avec une capacité énergétique limitée : Bluetooth, IEEE 802.11, HiperLAN2 et UWB sont les technologies les plus fréquemment utilisées.

Nous pouvons faire ressortir quatre caractéristiques (contraintes) essentielles qui doivent être prises en considération dans la conception de tout protocole dédié aux réseaux ad hoc :

- *Topologie dynamique*. Les nœuds se déplacent indépendamment les uns des autres et de manière imprévisible. Un nouveau terminal peut intégrer le réseau et en ressortir à n'importe quel moment.
- *Capacités des liens limitées*. Même si les technologies sans fil ont fait d'énormes progrès en termes de bande passante, ils restent néanmoins inférieurs à ceux des technologies filaires. La bande passante varie d'un type de réseau ad hoc à un autre. Par exemple, les performances attendues dans les réseaux de capteurs ne seront pas les mêmes que celles attendues dans les réseaux ad hoc d'entreprise.
- *Autonomie et puissance de calcul limitées*. Comme nous l'avons vu un peu plus haut, les nœuds d'un réseau ad hoc sont de petits dispositifs électroniques qui, d'une part, fonctionnent en mode batterie dans la majorité du temps et, d'autre part, sont limités en puissance de calcul. Il faut aussi noter que ces deux caractéristiques dépendent l'une de l'autre.
- *Protection physique*. Les nœuds d'un réseau ad hoc n'ont pas la même protection physique que les nœuds d'un réseau filaire. En effet, ceux d'un réseau ad hoc sont censés être mobiles et parfois complètement autonomes, c'est notamment le cas des réseaux de capteurs où les nœuds sont souvent lâchés, dans un environnement particulier et parfois hostile, sans aucune surveillance particulière.

L'intérêt principal des réseaux ad hoc est leur capacité à réduire le coût de déploiement. De plus, ils permettent des communications de proche en proche dans des environnements où il n'est pas possible d'installer une infrastructure complète pour des raisons techniques ou fonctionnelles. Après une configuration initiale du terminal, les réseaux ad hoc sont généralement auto-administrés, c'est-à-dire que les utilisateurs eux-mêmes découvrent et

utilisent les ressources disponibles dans leur voisinage, communiquent entre eux, partagent des données, des applications, etc., sans avoir besoin d'une administration centralisée. Néanmoins, si ce fonctionnement est très appréciable dans un environnement public, il peut représenter un problème dans un contexte sécuritaire.

Cette description correspond à un réseau purement ad hoc. En pratique il existe une infinité de variantes mélangeant ad hoc et infrastructure. En effet, selon l'application visée et selon les contraintes liées à l'environnement de déploiement du réseau, on aura des réseaux privilégiant l'approche ad hoc ou l'approche infrastructure.

Voici quelques exemples de réseaux purement ou partiellement ad hoc utilisés dans différents domaines.

Réseaux mesh : Les opérateurs d'un réseau à infrastructure peuvent avoir la possibilité d'étendre la couverture de leur réseau à un moindre coût et ainsi accroître le nombre de clients potentiels. Par exemple, un mobile trop éloigné du réseau de l'opérateur (cellule GSM, point d'accès Wi-Fi, etc.) peut bénéficier du réseau ad hoc et du routage du trafic assuré par les nœuds ad hoc pour communiquer avec l'infrastructure.

Réseaux *VANETs* (pour *Vehicular Ad hoc Networks*) : Dans ces réseaux, le but est de fournir de l'information à des véhicules sur les autoroutes. Lorsqu'ils sont suffisamment proches les uns des autres, les véhicules constituent automatiquement un réseau. Ainsi, dès que l'un de ces véhicules est près d'un point d'accès, il commence à recevoir de l'information qu'il diffuse par la suite aux autres véhicules du même réseau ad hoc.

Réseaux de capteurs : Un réseau de capteurs peut être composé d'une centaine ou de milliers de nœuds. Ces appareils, appelés en anglais *motes*, sont des dispositifs électroniques pas très coûteux et limités en ressources : énergie, mémoire et capacité de calcul. Ils sont déployés de façon aléatoire dans des environnements souvent ouverts. Dans la majorité des cas, les capteurs font, indépendamment des uns des autres, des mesures périodiques et envoient les données collectées à un dispositif plus puissant appelé le puits (sink) ou la station de base, qui agrège les données reçues en calculant, par exemple, leur maximum, moyenne ou médiane.

On peut aussi classer les réseaux ad hoc dans la catégorie des réseaux autonomes et spontanés. Cette catégorie inclut également les réseaux pair-à-pair. En effet, l'une des caractéristiques essentielles d'un réseau autonome est de parvenir à offrir des services en

s'auto-configurant et en coopérant directement sans l'intervention d'une quelconque partie extérieure.

2.2 *Le routage dans les réseaux ad hoc*

La fonction de routage est l'une des plus importantes dans les réseaux de communication, que ce soit dans les réseaux filaires ou dans les réseaux sans fil. Le routage sert à faire communiquer des nœuds distants. C'est un mécanisme qui permet l'établissement et le maintien de routes optimales entre les nœuds d'un réseau. Suivant les besoins et la finalité du réseau, une route optimale est choisie selon différents critères. Il existe des protocoles de routage classiques qui s'appuient sur la longueur de la route afin de choisir la meilleure d'entre elles (c'est-à-dire, la route optimale est celle qui a le plus petit nombre de sauts) (par exemple [PBRD03, PB94, OLS03, DBDAJ01]). D'autres protocoles prennent en considération d'autres critères comme la qualité de service (par exemple [LL99]). On retrouve aussi des protocoles de routage qui s'appuient sur des considérations de fiabilité et de sécurité afin de choisir les nœuds qui vont composer une route (par exemple [MM02, BLB02]).

Il existe des protocoles de routage conçus pour les réseaux avec infrastructure (tels que les réseaux filaires) (par exemple [Hed88, RL95]), ou sans infrastructure (tels que les réseaux ad hoc) (par exemple [PBRD03, PB94, OLS03, DBDAJ01]). Cependant, leurs caractéristiques sont complètement différentes. Les protocoles de routage destinés aux réseaux filaires n'ont pas besoin de gérer les contraintes liées à la mobilité. Ces protocoles ne se focalisent pas non plus sur la minimisation des communications sachant que les réseaux filaires sont réputés pour être des réseaux à haut débit. Un autre point très important est le fait que dans les réseaux filaires, le routage est réalisé par des nœuds de confiance appelés routeurs. Ces caractéristiques changent radicalement dès lors qu'il s'agit des réseaux sans fil ad hoc. En effet, la mobilité qui est une caractéristique de base des réseaux ad hoc, les contraintes liées à la limitation de ressources (bande passante, mémoire, processeur, contraintes d'énergie) et l'absence d'une entité de confiance, ont contraint les chercheurs à concevoir des protocoles de routage spécifiques aux réseaux sans fil ad hoc.

Avant de détailler le fonctionnement des différents protocoles de routage dans les réseaux ad hoc, nous allons tout d'abord présenter leurs différentes catégories. Il existe deux façons de classifier les protocoles de routage dans les réseaux ad hoc. La première classification se base sur la manière dont les informations de routage sont structurées et échangées dans le réseau. Cette classification donne lieu à deux classes de méthodes :

- la méthode du vecteur de distance ;
- la méthode d'état des liens.

Dans la méthode du vecteur de distance, chaque nœud du réseau maintient une table de routage contenant les distances vers tous les nœuds du réseau. À chaque fois qu'un nœud reçoit une mise à jour de routage (table de routage) de la part d'un nœud voisin, il l'examine afin de déduire des routes vers de nouvelles destinations ou bien pour mettre à jour des routes déjà existantes dans sa table de routage. Une fois que ses propres informations de routage sont mises à jour, il les envoie à ses voisins, qui à leur tour font la même chose et ainsi de suite. Un exemple d'un protocole de routage appartenant à cette catégorie est le protocole DSDV proposé dans [PB94]. Un inconvénient des protocoles basés sur ce principe est qu'ils convergent très lentement.

Les protocoles basés sur le routage par état de liens essaient de corriger les limitations des protocoles du vecteur de distance. Le fonctionnement de ces protocoles peut être résumé ainsi : chaque nœud du réseau découvre son voisinage (les identités des nœuds à un saut) en diffusant périodiquement des messages appelés messages *hello* que tous ses voisins potentiels peuvent intercepter ; une fois que le nœud a découvert son voisinage, il transmet aux nœuds du réseau un message contenant la liste de tous ses voisins ainsi que le coût nécessaire pour atteindre chacun d'entre eux ; ces messages sont enfin utilisés par les nœuds du réseau afin d'avoir une vue globale de la topologie du réseau. Le protocole OLSR [OLS03] est l'exemple type des protocoles utilisant ce principe. Cette approche permet une convergence plus rapide que par la méthode du vecteur de distance.

Une deuxième méthode de classification des protocoles de routage peut être obtenue en se basant sur le déclenchement des mises à jour des informations de routage. De cette classification il ressort deux approches, l'approche réactive et l'approche proactive.

2.2.1 Protocoles proactifs

La particularité de l'approche proactive est que les routes sont construites de façon continue même si elles ne sont pas utilisées dans l'immédiat. Nous allons détailler ici deux exemples de protocole de cette catégorie à savoir OLSR et DSDV.

DSDV :

Le protocole DSDV (Destination Sequenced Distance Vector Protocol) [PB94] est une adaptation du protocole RIP (Routing Information Protocol) [Hed88] aux réseaux ad hoc.

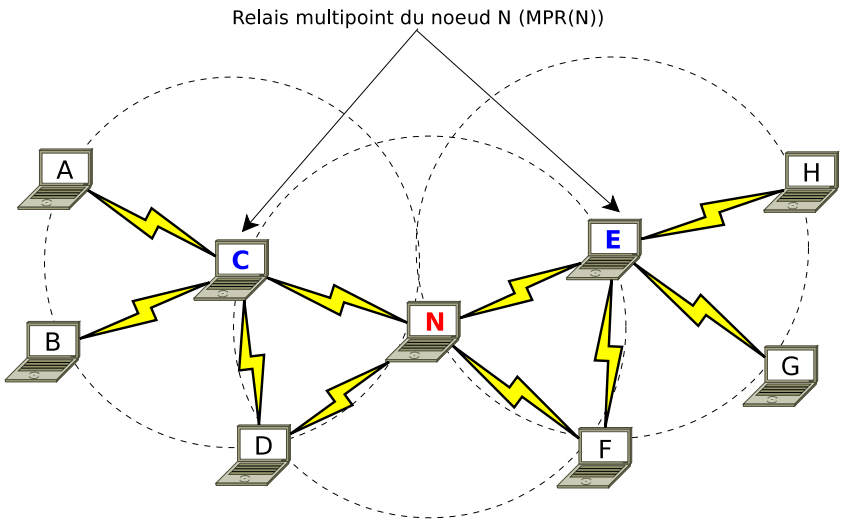

FIGURE 2.5 – Optimisation des relais multipoint.

Les concepteurs de ce protocole ont rajouté un nouvel attribut (le numéro de séquence) aux entrées de la table de routage qui permet à chaque nœud de distinguer entre les anciennes et nouvelles routes et ainsi éviter la formation des boucles de routage. Dans DSDV, la structure d'une ligne de la table de routage est définie comme suit : < Destination, Noeud suivant, Nombre de sauts, Numéro de séquence>. Afin de maintenir la cohérence des tables de routage, chaque nœud diffuse périodiquement ou lors d'un changement de topologie des paquets de mise à jour (tables de routage) à ses voisins. À la réception par un nœud d'un paquet de mise à jour, il compare cette information à celles qui existent déjà dans sa table de routage et transmet à son tour un paquet de mise à jour à ses voisins. Si plusieurs paquets de mise à jour sont reçus par un nœud avec des numéros de séquence différents pour la même destination, alors la route avec le plus grand numéro de séquence est sélectionnée. Par contre, si ces paquets de mises à jour ont le même numéro de séquence, alors c'est la route avec la distance (nombre de sauts) la plus courte qui est choisie. En plus d'être lent à converger, le protocole DSDV génère un surcoût considérable en termes de communication ($O(n^2)$ pour un réseau de n nœuds), ce pourquoi il n'est considéré viable que dans des environnements très particuliers (réseaux relativement statiques et de petite taille).

OLSR :

OLSR (Optimized link State Routing Protocol for Ad-Hoc Networks) [OLS03], est un protocole de routage de nature proactive basé sur un algorithme d'état de liens. Il emploie des échanges périodiques de messages pour maintenir les informations sur la topologie du réseau. C'est une optimisation de la méthode d'état des liens, du fait qu'il utilise des messages compacts et qu'il limite le nombre de retransmissions lors de la diffusion des paquets de contrôle. À cet effet, il se sert de relais multipoint afin de rendre la diffusion des messages de contrôle plus précise et économique. Les relais d'un nœud N sont un sous-ensemble de ses voisins à un saut. Cet ensemble, qui est appelé MPRs du nœud N et noté $MPR(N)$, est choisi de telle façon que les liens avec ces voisins soient bi-directionnels et que tous les nœuds à deux sauts de N soient atteints. Les nœuds n'appartenant pas à l'ensemble MPR du nœud N peuvent lire et traiter les paquets reçus du nœud N, mais ne doivent pas les retransmettre. La figure 2.5 illustre le mécanisme d'optimisation des relais multipoint. La retransmission par les nœuds C et E est suffisante pour assurer la diffusion d'un paquet en provenance de la station N. L'ensemble des stations C et E forme un ensemble de relais multipoint du nœud N. Une diffusion issue du nœud N va donc utiliser uniquement trois transmissions.

2.2.2 Protocoles réactifs

Les protocoles de routage appartenant à cette catégorie, créent et maintiennent les routes selon leurs besoins. Lorsque le réseau a besoin d'une route, une procédure de découverte globale de routes est lancée, et ce, dans le but d'obtenir une information spécifique. Dans ce qui suit, nous allons décrire les protocoles les plus importants de cette classe :

DSR

Le protocole DSR (Routage à Source Dynamique) [DBDAJ01], est basé sur l'utilisation de la technique "routage source". Dans cette technique, la source des données détermine la séquence complète des nœuds à travers lesquels les paquets de données seront envoyés. Un nœud initiateur de l'opération de découverte de routes diffuse un paquet de requête de route dans l'ensemble du réseau. Si l'opération de découverte est réussie, l'initiateur reçoit un paquet réponse de route qui liste la séquence de nœuds à travers lesquels la destination peut être atteinte. Le paquet réponse de route contient donc un champ contenant la séquence des nœuds visités durant la propagation de la requête dans le réseau. Afin d'assurer la validité des chemins utilisés, le protocole DSR exécute une procédure de maintien de routes :

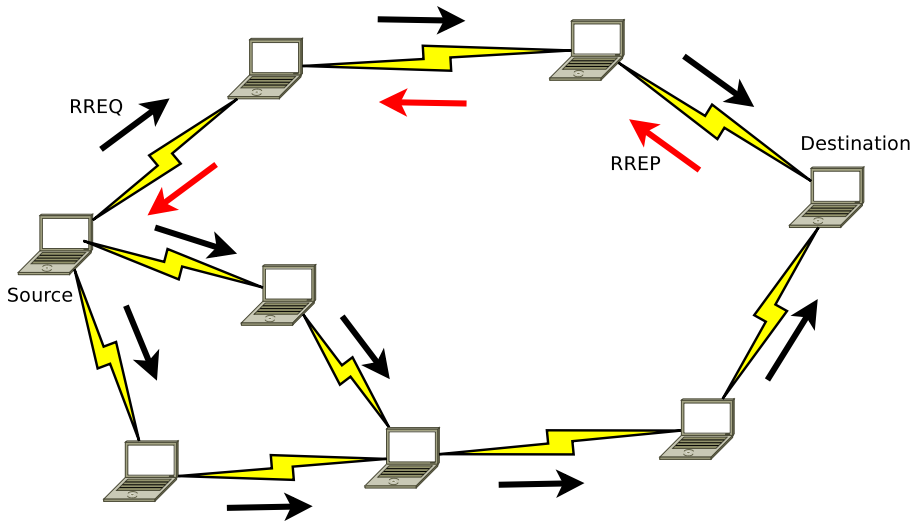

FIGURE 2.6 – Procédure de découverte de route du protocole AODV

- quand un nœud détecte un problème dans la transmission (rupture de liens), un message *erreur de route* est envoyé à l'émetteur original du paquet ;
- le message d'erreur contient l'adresse du nœud qui a détecté l'erreur et celle du nœud qui le suit dans le chemin ;
- lors de la réception du paquet *erreur de route* par le nœud source, le nœud concerné par l'erreur est supprimé du chemin sauvegardé, et tous les chemins qui contiennent ce nœud sont tronqués à ce niveau là. Par la suite, une nouvelle opération de découverte de routes vers la destination est initiée par l'émetteur.

AODV

Le protocole AODV [PBRD03] s'inspire des protocoles DSDV et DSR. Il réduit le nombre de messages diffusés en créant les routes quand cela est nécessaire, contrairement au protocole DSDV qui maintient la totalité des routes. Ceci a pour effet de limiter la répercussion des modifications topologiques aux seules routes en cours d'utilisation.

AODV utilise le principe des numéros de séquence afin de maintenir la cohérence des informations de routage et ainsi utiliser les routes les plus récentes, car, à cause de la mobilité des nœuds dans le réseau, les routes changent fréquemment, ce qui rend les routes maintenues par certains nœuds invalides.

De la même manière que dans DSR, AODV utilise une requête de demande de route dans le but de créer un chemin vers une destination donnée. Cependant, AODV maintient les chemins d'une façon distribuée en gérant une table de routage au niveau de chaque nœud appartenant au chemin recherché. Un nœud diffuse une requête de route *RREQ* vers un nœud destination si :

- la destination n'est pas connue au préalable, ou
- la durée de vie du chemin existant vers la destination a expiré, ou le chemin est devenu défaillant.

Les nœuds qui reçoivent ces paquets les diffusent à leur tour jusqu'à atteindre la destination ou au moins un nœud qui possède une information de routage récente vers la destination recherchée. Les nœuds situés sur le parcours de la requête conservent, dans leur cache, l'adresse du nœud qui leur a relayé la requête. L'adresse de ce nœud fournit l'adresse du saut suivant vers le nœud source. Cette information est utilisée lors du retour de la réponse de demande de route *RREP* pour permettre d'aiguiller cette réponse jusqu'au nœud source.

Le champ *numéro de séquence destination* du paquet *RREQ*, contient la dernière valeur connue du numéro de séquence associé au nœud destination qui est recopiée de la table de routage. Si le numéro de séquence n'est pas connu, la valeur nulle sera prise par défaut. Le numéro de séquence source du paquet *RREQ* contient, quant à lui, la valeur *du numéro de séquence du nœud source*.

Afin de maintenir l'état correct des routes, une transmission périodique d'un message *hello* est effectuée. Si, par exemple, trois messages *hello* consécutifs ne sont pas reçus à partir d'un nœud voisin, le lien en question est considéré comme étant défaillant et un message d'erreur *RERR* est envoyé à chaque source concernée par ce lien afin de lui permettre de mettre à jour ses informations de routage en relançant de nouvelles procédures de recherche de routes.

2.2.3 Protocoles réactifs Vs Protocoles proactifs

Les protocoles proactifs possèdent de bonnes aptitudes en termes de temps de réponse, car quand un nœud souhaite communiquer avec un autre nœud, il trouve immédiatement dans sa table de routage la route adéquate pour atteindre ce nœud. Cependant, un surcoût significatif est engendré par les échanges périodiques de messages de contrôle. Les protocoles réactifs résolvent plus ou moins le défaut des protocoles proactifs tout en

augmentant le temps de réponse. Le surcoût engendré en utilisant l'approche réactive est moindre dans certaines configurations réseau. Néanmoins, il peut demeurer élevé dans d'autres. De manière générale, les protocoles réactifs et proactifs présentent des performances différentes selon les caractéristiques du réseau. Dans le cas d'un réseau dense par exemple et/ou lorsque l'échange des données est intense, le choix d'un protocole proactif s'avère plus judicieux, car un protocole réactif entraîne la diffusion excessive de messages de demande de routes qui risquent de saturer le réseau. En revanche, dans une configuration opposée, à savoir un réseau à densité faible et/ou à faible trafic, choisir un protocole réactif est plus intéressant, car il ne surchargera pas inutilement le réseau par des flux continuels d'informations de routage entre les nœuds.

2.2.4 Protocoles hybrides

Afin de tirer profit des avantages des protocoles réactifs et proactifs, des protocoles qui mêlent généralement les deux modes (fonctionnant dans l'un ou dans l'autre mode) suivant des conditions prédéfinies, ont été proposés.

Dans ce type de protocoles, on utilise une reconnaissance locale de la topologie réseau jusqu'à un nombre relativement petit de sauts, par un échange périodique de trames de contrôle, autrement dit par une technique proactive. Les routes vers des nœuds plus lointains sont obtenues par un schéma réactif, c'est-à-dire par l'utilisation de paquets de demande de routes qui sont dans la majorité des cas diffusés dans tout le réseau.

Parmi les protocoles hybrides, nous nous intéressons particulièrement à *ZRP* (*Zone Routing Protocol*). *ZRP* a été introduit en 1997 par Haas et Pearlman [Haa97, HP01].

ZRP définit pour chaque nœud une zone de routage (*zone radius*), qui inclut tous les nœuds dont la distance minimale (en nombre de sauts) à ce nœud est d. Les nœuds qui sont exactement à distance d sont appelés nœuds périphériques. Pour trouver une route vers des nœuds situés à une distance supérieure à d, *ZRP* utilise un système réactif, qui envoie une requête à tous les nœuds périphériques. Pour cela, *ZRP* met en œuvre deux types de fonctionnement : *IARP* (IntrAzone Routing Protocol) et *IERP* (IntErzone Routing Protocol).

Basé sur un protocole à vecteur de distance, *IARP* permet, en utilisant une technique proactive, de trouver toutes les routes jusqu'à une distance d. *IERP* quant à lui, permet d'établir les routes vers les nœuds à plus de d sauts d'une façon réactive.

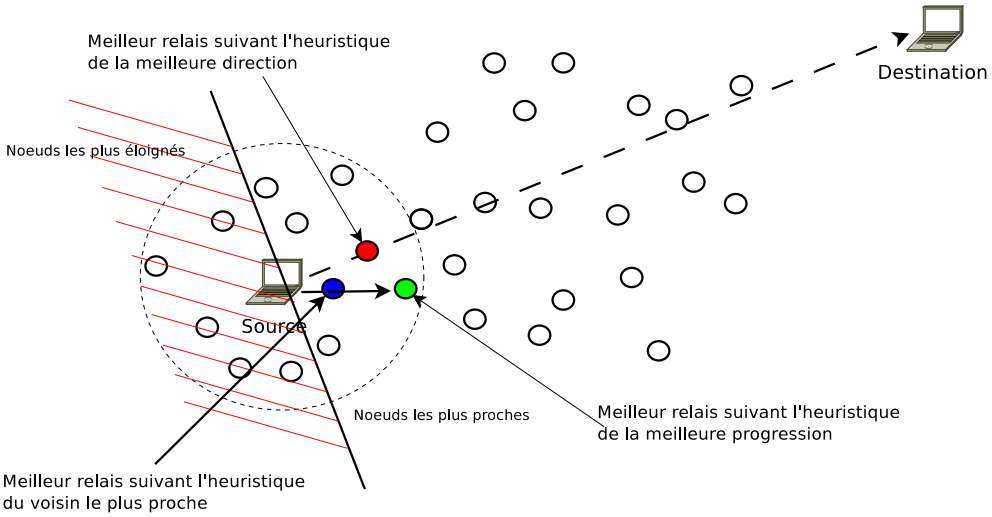

FIGURE 2.7 – Routage géographique

Le processus de recherche de route fonctionne de la façon suivante :

- si une route est connue, cela signifie que la destination est à moins de d sauts. Cette route est la route retournée par le protocole ;
- si aucune route n'est connue, cela signifie que le nœud est situé à une distance supérieure à d. Dans ce cas, le nœud envoie une requête par *IERP* à tous les nœuds périphériques ;
- si un nœud périphérique a connaissance d'une route disponible, il renvoie une réponse. Dans le cas contraire, le protocole se poursuit récursivement jusqu'à obtention d'une route.

2.2.5 Protocoles de routage géographiques

L'idée de ce type de protocoles [KKoo] est d'utiliser des informations géographiques pour acheminer les paquets. Pour simplifier la présentation, nous supposerons que tous les nœuds connaissent leur position et que ceux-ci connaissent également la position de tous les autres nœuds du réseau. Avec ces hypothèses, il est facile de concevoir qu'un nœud peut facilement choisir parmi ses voisins un relais pour acheminer un paquet dont il connaît la destination finale et donc aussi sa position. Il est très simple d'envisager des critères de sélection parmi ces voisins ; donnons quelques exemples parmi les heuristiques les plus classiques. Une

heuristique particulièrement simple est celle qui choisit le voisin qui est le plus proche de la direction du destinataire final. Une autre heuristique proche bien que différente est celle qui choisit le voisin qui permet de progresser le plus en direction du destinataire final, c'est l'heuristique de la progression maximale. Cette dernière heuristique est intéressante, car on peut montrer qu'en moyenne elle permet de minimiser le nombre de sauts vers la destination. Enfin, une autre heuristique très simple est celle qui consiste à choisir le voisin le plus proche en direction de la destination finale. Il est possible de montrer que cette heuristique permet de minimiser l'énergie locale pour relayer le paquet vers sa destination finale.

Il est important de noter qu'il existe des protocoles basés sur des approches radicalement différentes de celles que nous avons décrites, comme c'est le cas des protocoles basés sur le fonctionnement des colonies de fourmis. Ces protocoles n'ont pas de nos jours remplacé les approches plus traditionnelles que nous avons décrites ci-dessus, mais forment un axe de recherche très dynamique et prometteur. Nous avons réalisé des travaux sur ces protocoles qui ont donné lieu à une publication dans un workshop international [AAGT08]. Cependant, vu que ceci est en marge du sujet de ce manuscrit, nous avons décidé d'introduire ces protocoles de façon générale ainsi que nos travaux en annexe de ce document (c.f. Annexe A).

2.3 *Menaces, attaques dans les réseaux ad hoc*

Après avoir présenté les notions de base relatives aux réseaux sans fil ad hoc, abordons maintenant les différentes menaces, attaques et vulnérabilités qu'on peut retrouver dans ce type de réseaux.

L'utilisation des liens sans fil facilite considérablement les attaques contre les réseaux ad hoc, que ce soit à travers de simples écoutes ou d'attaques plus nuisibles comme le dysfonctionnement intentionnel d'un service (déni de service). Contrairement aux réseaux filaires où les attaquants doivent avoir un accès physique au réseau ou bien outrepasser plusieurs couches de défense telles que les pare-feu et les passerelles, dans les réseaux sans fil, il suffit que l'attaquant soit dans le champ de transmission d'un nœud pour pouvoir intercepter les communications de ce dernier. Enfin, dans certains contextes comme celui des réseaux de capteurs, les nœuds ont une faible protection physique et peuvent donc être capturés, corrompus et détournés à des fins malfaisantes.

La majorité des réseaux autonomes et auto-organisés, et tout particulièrement les réseaux sans fil ad hoc, s'appuient sur des algorithmes coopératifs nécessitant l'implication de tous les nœuds du réseau, et ce, afin d'assurer le fonctionnement de la majorité des services. Prenons l'exemple des protocoles de routage qui sont plus vulnérables dans les réseaux ad hoc que dans les réseaux avec infrastructure, car chaque nœud doit faire office de routeur pour ses voisins. Des nœuds pourraient par exemple décider de supprimer ou de modifier les informations de routage qu'ils reçoivent de leurs voisins ou bien injecter de fausses informations de routage. Ces comportements peuvent s'avérer préjudiciables pour le bon fonctionnement du réseau dans la mesure où le routage constitue l'un des noyaux de base du fonctionnement des réseaux informatiques. Un autre comportement malveillant, plus facile à réaliser, mais tout aussi néfaste pour le bon fonctionnement du réseau est le comportement dit *égoïste* d'un nœud [MGLBoo]. Autrement dit, certains nœuds pourraient refuser de router les paquets des autres afin de préserver leurs ressources matérielles.

2.3.1 Modèles d'attaquant

En règle générale on peut classer un attaquant suivant plusieurs critères : interne ou externe selon son appartenance au réseau ; global ou local selon sa portée ; actif ou passif selon son comportement.

Interne/Externe Un attaquant interne est celui qui arrive à contrôler un nœud ayant le statut de membre à part entière du réseau et qui dispose donc de l'ensemble de connaissances associées à ce statut (clés secrètes, table de routage, etc.). Par opposition, un attaquant externe ne dispose pas a priori de ces connaissances. En particulier, il est incapable d'effectuer des opérations cryptographiques comme signer, déchiffrer, etc. L'utilisation de systèmes de chiffrement et de signature pour sécuriser les communications est un moyen efficace de restreindre les actions d'un attaquant externe. En effet, dans un tel cadre, les actions disponibles pour l'attaquant vont se borner à détecter la présence/absence d'une communication et éventuellement à supprimer des paquets, ce qui limitera fortement l'éventail des attaques qu'il pourra réaliser. Cependant, ceci introduit un fort coût qui peut être considéré comme difficilement acceptable dans un réseau ad hoc. Ainsi, en fonction du contexte, on ne pourra pas du tout utiliser ces techniques cryptographiques ou tout au plus on ne pourra les utiliser que pour une sous-partie des communications ouvrant de nouvelles options pour ces attaquants.

Global/Local Généralement, un attaquant, va avoir un rayon d'action limité autour d'un

certain nombre de points (que ce soit par un ou plusieurs nœuds du réseau qu'il contrôle ou par un dispositif autre). On va dans ce cas parler d'un attaquant local. On considère souvent un modèle plus contraignant, celui d'un attaquant qui posséderait le don d'ubiquité dans le réseau. Cet attaquant, dit global, est dans beaucoup de cas peu réaliste, mais permet de s'affranchir de certaines considérations comme par exemple à quel point un attaquant local peut choisir convenablement ses points d'observation/action pour mener une attaque. Si on résiste à un attaquant global, on résiste aussi à un attaquant local quel que soit son choix des points d'attaque. Bien sûr résister à un attaquant global est contraignant et de nombreux protocoles résistant à un attaquant local donné vont se baser sur des hypothèses plus faibles comme le fait qu'il n'y ait pas un choix évident d'un nombre restreint de points d'attaque permettant de briser le système.

Actif/Passif Un attaquant passif va se restreindre à écouter les flux de communication et essayer uniquement par ces écoutes d'obtenir et stocker des informations qu'il n'est pas supposé connaître ou garder. Face à un tel attaquant, le but va généralement être d'empêcher qu'il puisse obtenir des données à caractère privatif. Les protocoles de base tels que le routage, mais aussi les protocoles essayant de sécuriser le réseau, comme par exemple les systèmes de confiance, vont dévoiler de nombreuses informations qui peuvent être utilisées pour obtenir des données privatives d'un nœud ou d'un ensemble de nœuds. Face à une telle menace, l'objectif va généralement être de développer de nouveaux protocoles diffusant un minimum d'informations privatives tout en essayant de garantir les mêmes propriétés et la même efficacité que les protocoles traditionnels. Un attaquant actif quant à lui pourra supprimer, modifier ou injecter des paquets, et de façon plus générale modifier son comportement de façon arbitraire par rapport au comportement normal dans les différents protocoles dans lesquels il sera engagé. Un tel attaquant pourra ainsi non seulement obtenir plus d'informations qu'un attaquant passif, mais aussi il pourra modifier significativement le fonctionnement d'un protocole ou en empêcher son exécution. Dans un environnement collaboratif comme celui des réseaux ad hoc, un tel attaquant peut en pratique être extrêmement nuisible, en particulier s'il contrôle un ou plusieurs nœuds du réseau. Afin de réduire la marge de manœuvre d'un tel attaquant, il est possible de mettre en place des mécanismes visant à contraindre un tel attaquant à des actions non-observables. Pour des actions ayant un caractère inéquivoquement malveillant, des mécanismes d'exclusion peuvent être mis en place, mais une réaction aussi irrévocable n'est souvent pas très polyvalente et peut introduire des nouvelles failles de sécurité (permettant des accusations à tort de nœuds bienveillants). Les mécanismes de confiance ou de réputation

permettent d'établir une réponse graduée et efficace contre certains attaquants actifs, mais aucun système ne permet de sécuriser un réseau ad hoc contre les attaques actives de façon générale, en particulier si on considère des attaquants internes ou des attaquants ne craignant pas des représailles.

2.3.2 Description des principales attaques

Dans cette section, nous décrivons dans un premier temps quelques attaques spécifiques au routage dans les réseaux ad hoc. Nous présentons dans un deuxième temps les attaques de type Sybille, celles-ci ayant un impact important sur nos travaux.

2.3.2.1 Attaques au niveau du routage

La fonction de routage est essentielle au bon fonctionnement d'un réseau. Malheureusement, le caractère collaboratif des réseaux ad hoc ainsi que le fait que tous les nœuds aient le même niveau d'autorité, rendent plus facile une attaque contre cette fonction. Toutes les attaques que nous verrons, ici, sont des attaques actives opérant au niveau des différentes phases du routage :

attaque du trou noir (Black Hole Attack). Dans ce cas de figure, un nœud malhonnête tente d'exploiter les failles des protocoles de routage afin de se faire élire comme faisant partie du plus court chemin vers le nœud dont il veut intercepter les paquets. Il pourra donc recevoir les paquets destinés à ses victimes et les supprimer afin de réaliser un déni de service ou bien utiliser son positionnement sur ces routes et lancer par exemple les premières étapes d'une attaque de l'homme du milieu.

attaque par usurpation d'identité (Identity Spoofing). Un nœud peut usurper l'identité d'un autre nœud du réseau. Il peut ainsi recevoir les paquets destinés aux nœuds légitimes, diffuser de fausses informations de routage (envoyer par exemple de faux paquets de contrôle à la place du nœud légitime), etc.

attaques par modification des paquets de contrôle. Un nœud peut modifier de façon illégale les paquets de contrôle envoyés par les autres nœuds du réseau dans le but de les tromper. Les modifications peuvent concerner plusieurs informations de routage telles que les numéros de séquence ou le nombre de sauts.

Noeuds égoïstes. Des nœuds peuvent ne pas vouloir participer aux processus d'établissement des routes et décider de ne pas relayer les paquets de contrôles reçus par souci d'économie de batterie ou d'utilisation du CPU.

Attaque Wormhole (attaque du trou de ver). Deux nœuds se trouvant généralement dans des zones géographiquement séparées vont former une collusion afin de transporter les paquets de contrôle des autres au-delà du périmètre autorisé. Ce qui nuit sérieusement à la fonction de routage. Le détail de cette attaque est décrit dans les paragraphes qui suivent.

Attaques par déni de service. L'attaquant peut simplement perturber le fonctionnement d'un protocole de routage. L'attaquant peut, par exemple, envoyer de fausses informations de routage : envoyer de faux paquets de contrôle (p. ex. de faux paquets d'erreur RtERR), mentir sur quelques métriques de routage (p. ex. nombre de sauts dans AODV, la liste des prochains sauts dans DSR), construire des boucles de routage, falsifier les tables de routage, etc.

Rushing attack. Cette attaque s'appuie sur le fonctionnement des protocoles de routage réactifs et plus précisément sur la diffusion des paquets de demande de route. Dans ces protocoles, les nœuds ne tiennent compte que du premier paquet reçu et suppriment toutes les autres copies d'une même demande de route. Dans l'attaque Rushing, l'attaquant exploite cette propriété en devançant les autres nœuds dans l'envoi des paquets de demande de route. Cela est possible en ne respectant pas, par exemple, les délais de transmission des paquets de demande de route tels qu'ils ont été spécifiés par le protocole de routage ou bien en ne respectant pas les délais d'accès aux médias de communication. Un des moyens de contrer cette attaque est de permettre aux nœuds de sélectionner de manière aléatoire la copie de demande de route à relayer. Autrement dit, chaque nœud attend une période de temps t avant de sélectionner aléatoirement une copie d'une demande de route reçue durant cette période de temps.

L'attaque Wormhole a tout particulièrement éveillé l'intérêt de la communauté scientifique, autant par son originalité que par la difficulté de trouver des contre-mesures ayant un coût raisonnable dans un réseau ad hoc. Aussi, nous présentons cette attaque plus en détail dans le reste de cette section.

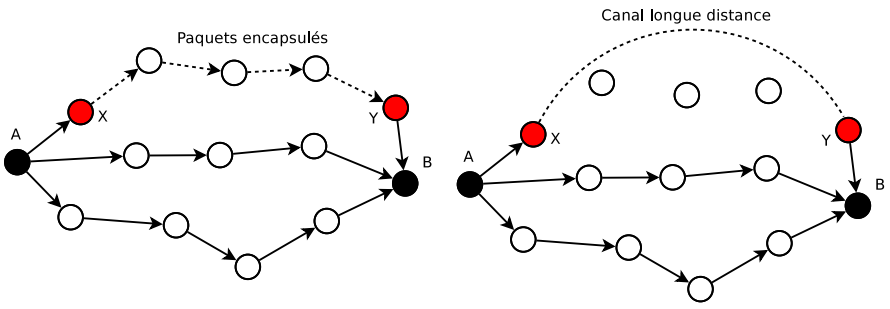

FIGURE 2.8 – Attaque Wormhole

Attaque Wormhole

L'attaque du trou de ver est une attaque particulièrement difficile à contrer. Elle peut être lancée par un attaquant externe et être réussie même en présence d'un système d'authentification et de chiffrement.

Une version simple de cette attaque est de faire croire à deux de ses voisins qu'ils sont eux aussi voisins en relayant leurs paquets. Une version plus élaborée de cette attaque nécessite la présence d'au moins deux nœuds malveillants formant une collusion et se trouvant généralement dans des zones géographiquement séparées. Le but de cette version de l'attaque est de former un tunnel entre ces deux nœuds. Ce tunnel est construit de telle sorte que chaque paquet capturé à l'une des extrémités du tunnel soit relayé, à travers ce tunnel, vers le deuxième attaquant se trouvant à l'autre bout du tunnel. Celui-ci à son tour les relaie localement, selon le type du paquet reçu, à ses voisins. Il existe plusieurs façons de construire un tunnel, considérons ces deux méthodes (voir figure 2.8) :

- les paquets capturés par un des nœuds malveillants sont chiffrés et envoyés à travers une route, préalablement établie par le protocole de routage utilisé, vers le deuxième nœud malveillant distant, qui à son tour déchiffre les paquets reçus et les relaie à ses voisins ;
- les deux attaquants sont supposés être reliés par une liaison rapide (p. ex. liaison filaire ou une liaison sans fil longue distance).

Cette attaque engendre des effets très négatifs sur les protocoles de routage et donne un

moyen simple aux intrus de faire partie de n'importe quelle route active. Prenons l'exemple de la figure 2.8. Les paquets *hello* émis par un nœud A sont capturés par une des extrémités du tunnel X et envoyés vers l'autre extrémité du tunnel Y. Dans ce cas de figure, si le nœud B diffuse, à son tour, le message *hello* à ses voisins, ces derniers seront induits en erreur quant à l'appartenance du nœud A à leur voisinage.

Un autre exemple est celui d'une demande de route acheminée de la même manière que le paquet *hello* précédent. Dans ce cas, le paquet de demande de route arrive au nœud Y avec un nombre de sauts (dans le cas d'un protocole de routage basé sur cette métrique d'évaluation de la meilleure route) très inférieur à ceux des paquets de demande de route envoyés par l'intermédiaire d'autres chemins. Ainsi, une route passant par le trou de ver sera beaucoup plus intéressante dans de nombreux cas et permettra aux attaquants de garder à leur portée le trafic qu'ils souhaiteront.

Plusieurs solutions de prévention ont été proposées afin de remédier à cette attaque [HPJ03, HE04].

Dans [HPJ03], les auteurs préconisent de rajouter à chaque paquet quelques informations authentifiées, permettant de limiter la distance qu'un paquet a le droit de parcourir. A cet effet, ils proposent d'utiliser deux types d'informations : les positions géographiques (les nœuds doivent être capables de déterminer de manière sûre leurs positions géographiques) ou bien des informations temporelles (p. ex. l'heure à laquelle les paquets sont transmis ou bien un temps d'expiration après lequel les paquets doivent être supprimés). Les informations géographiques assurent que le récepteur d'un paquet est à une certaine distance de l'expéditeur et les informations temporelles, efficaces uniquement dans le cas de la première forme de l'attaque wormhole, permettent quant à elles que la durée de vie des paquets dans le réseau soit bornée, ce qui limite par conséquent les distances que les paquets peuvent parcourir. Ces méthodes sont difficiles à réaliser en pratique à cause des problèmes de synchronisation et d'accès au médium de communication liés aux réseaux ad hoc.

Une autre approche de prévention de l'attaque du trou de ver est donnée dans [HE04]. Les auteurs proposent une méthode originale basée sur le fonctionnement des antennes directionnelles. Grâce à la capacité des antennes à connaître la direction d'arrivée d'un paquet, elles peuvent authentifier et ainsi détecter tout paquet émis par un nœud non voisin. Par exemple, lorsqu'un nœud x diffuse un message *hello* à ses voisins, ces derniers doivent indiquer dans leurs réponses la direction par laquelle le message *hello* a été reçu. Le nœud x accepte les paquets reçus dans une direction opposée à celle déclarée dans ces paquets

et rejette tous les autres. Il faut noter que dans cette approche les communications entre voisins sont chiffrées et que les paquets sont signés. Cette première version ne permet pas de prévenir toutes les attaques du trou de ver et ce premier schéma est lui même vulnérable à l'attaque wormhole, pour cela les auteurs proposent une deuxième version qui permet le partage des informations (qui est le voisin de qui et par quelle direction chaque nœud reçoit les paquets de ses présumés voisins) entre nœuds voisins et augmente ainsi le pourcentage d'attaques évitées.

2.3.2.2 Attaque Sybille

L'attaque Sybille a tout d'abord été introduite et décrite par Douceur [Dou02] dans le contexte des réseaux pair-à-pair. Elle a été définie initialement comme le fait qu'un nœud malveillant (*Sybil node*) parvienne à posséder illégitimement plusieurs identités et simule un ensemble de nœuds associés à ces identités. Par la suite, plusieurs variantes sont apparues dans différentes situations. Elle est considérée comme une attaque très difficile à prévenir ou à détecter dans de nombreux contextes.

L'attaque Sybille est également efficace contre les algorithmes de routage, l'agrégation de données, les algorithmes de vote, la répartition équitable des ressources, la détection de nœuds malveillants, etc. Par exemple, pour attaquer un protocole de routage, l'attaque Sybille peut donner l'illusion à un nœud de posséder plusieurs chemins vers une destination alors que tous ces chemins traversent la même entité physique.

Afin de se prémunir contre l'attaque Sybille, il faudrait trouver le moyen de vérifier si l'identité d'un nœud est la seule réellement utilisée par son dispositif physique.

Douceur [Dou02] a proposé, dans le contexte des réseaux pair-à-pair, de tester si certaines identités possèdent moins de ressources qu'elles ne seraient supposées en avoir si elles étaient indépendantes. L'hypothèse implicite sur laquelle s'appuie cette vérification est que tous les nœuds ont la même quantité de ressources. Les ressources testées dans ce cas sont la mémoire, le processeur et la capacité de transmission d'un nœud.

Dans [NSSP04], Newsome et *al.* ont proposé d'utiliser une autre ressource pour les tests à savoir l'interface radio d'un nœud. Leurs hypothèses de départ stipulent que chaque nœud possède une seule interface radio et que cette interface radio est incapable d'émettre ou de recevoir des données sur plusieurs canaux simultanément. Si un nœud veut vérifier si un de ses voisins est une identité Sybille, il alloue à chacun de ses voisins un canal différent.

Il choisit ensuite aléatoirement un canal et diffuse un message. Si le voisin écoutant sur ce canal est légitime, il devrait recevoir le message et peut y répondre. Supposons qu'il existe parmi les voisins de ce nœud s nœuds Sybille. Dans ce cas de figure, la probabilité que le nœud n'obtienne aucune réponse sur le canal choisi et que de ce fait ce canal corresponde à une identité Sybille est de s/n. En répétant ce processus de vérification r fois, on obtient une probabilité de non détection de $(\frac{n-s}{n})^r$.

Une autre approche très intéressante et très prometteuse introduite dans Newsome et *al.* pour contrer l'attaque Sybille est la vérification des positions géographiques des capteurs. Dans cette approche, le réseau vérifie la position géographique de chaque nœud. Les identités Sybille peuvent être détectées, car elles occupent exactement la même position géographique que le nœud malveillant qui les a générées. Un attaquant mobile peut résister à cette attaque s'il parvient à utiliser ces identités Sybille dans des positions géographiques différentes en se déplaçant. Afin de contrer cette attaque, il est nécessaire de tester simultanément (tout dépend de la vitesse de mobilité de l'attaquant) toutes les positions.

D'autres solutions basées sur une tarification de la délivrance des identités peuvent être envisagées. On retrouve aussi des solutions basées sur les systèmes de distribution de clés symétriques, sur la vérification à distance du code d'un capteur (elle suppose que le code exécuté par un nœud malveillant est différent de celui exécuté par un nœud honnête), etc. (cf. Newsome et *al.*).

L'obtention de plusieurs identités peut être problématique si des clés publiques sont utilisées pour authentifier les nœuds. Dans ce cas, un attaquant devra obtenir un certain nombre de clés ce qui peut être extrêmement difficile à réaliser. Dans le cadre des réseaux de capteurs, ceci donne lieu à une situation assez intéressante, puisqu'un attaquant aura plutôt intérêt à utiliser une seule identité à de multiples reprises en déployant des répliques d'un nœud à différents endroits du réseau. Il s'agit de l'attaque *par réplication*. Bien que ces répliques aient la même identité, l'attaquant peut utiliser le fait que les capteurs n'ont qu'une vision locale pour que la réplication ne soit pas détectée. Pour cela, il est bien sûr nécessaire que l'attaquant pirate un nœud du réseau et obtienne ses clés cryptographiques, mais cela est considéré plausible. En effet, en raison des contraintes matérielles liées aux réseaux de capteurs, il est souvent difficile de mettre en œuvre une protection physique résistante.

Certaines techniques permettent de détecter de telles répliques en se reposant sur la

liaison entre l'identité du nœud avec sa position géographique. Dans le chapitre 4, nous présentons différentes approches en mettant en évidence leurs avantages et inconvénients.

2.4 Conclusion

Nous avons présenté dans ce chapitre les concepts de base des réseaux ad hoc en faisant ressortir leurs propriétés les plus importantes à savoir le manque de ressources matérielles, l'utilisation de liens sans fil, le manque de protection physique au niveau des nœuds, les changements fréquents de topologie ou la difficulté de mettre en place un système de confiance. Ces propriétés engendrent de nouvelles problématiques qui n'existaient pas dans les réseaux filaires traditionnels et représentent des contraintes supplémentaires dans la conception des services de base du réseau.

La plupart des solutions de sécurité actuelles sont basées sur des primitives cryptographiques (chiffrement, signature, etc.) pour assurer les propriétés de base de la sécurité (confidentialité, intégrité, non répudiation, etc.). Cependant, ces primitives sont souvent utilisées comme un moyen de prévention (par exemple, en limitant les accès des utilisateurs) et constituent souvent le premier rempart de sécurité du réseau. Elles interviennent dans toutes les couches du réseau : couche physique, routage, etc. Toutefois, quand l'attaquant est à l'intérieur du réseau et corrompt des nœuds légitimes afin qu'ils se comportent de manière malfaisante ou quand la cryptographie est considérée trop coûteuse, d'autres approches sont nécessaires. Les systèmes de confiance et de réputation, que nous présentons dans le prochain chapitre, sont des candidats intéressants dans ce contexte. Ils sont utilisés comme une couche de sécurité supplémentaire permettant de surveiller le comportement des nœuds du réseau. Nous étudierons leurs principes de base et en particulier nous aborderons leur sécurité du point de vue de la préservation des données privées.

Ces propriétés influent aussi sur la gestion des identités. Une des attaques les plus nuisibles contre ce service est l'attaque Sybille que nous avons introduite dans ce chapitre. Dans le cadre des réseaux de capteurs, un autre problème similaire apparaît, il s'agit de l'attaque par réplication de nœuds que nous étudierons dans le chapitre 4.

Chapitre 3

Protection de la vie privée dans les systèmes de confiance

Le but premier des mécanismes de sécurité est d'offrir une protection contre les utilisateurs malveillants. Une politique de sécurité va traditionnellement limiter les accès aux seuls utilisateurs autorisés (par des systèmes d'authentification, des systèmes de contrôle d'accès ou de chiffrement) et ainsi donner des garanties sur certaines propriétés de sécurité importantes (confidentialité, intégrité, disponibilité, etc.). Par exemple, dans un réseau ad hoc, des mécanismes de sécurité [Zap02, sec02, HJP02, WKvO04, PH02, HPJ05, HTR+04, RACM04, ACL+03] basés sur quelques primitives cryptographiques permettent d'assurer, le contrôle d'accès au réseau, la liaison entre l'identité d'un nœud et ses paramètres publics (à l'aide par exemple des systèmes d'authentification), et la confidentialité et l'intégrité des paquets de données et de contrôle en utilisant des primitives cryptographiques telles que le chiffrement et la signature numérique.

Si les outils cryptographiques permettent dans certains cas de garantir la confidentialité, l'intégrité et l'authenticité des messages, il ne permettent pas de faire face à de nombreux comportements malfaisants comme les écarts dans les protocoles établis, le refus de collaborer, le non respect des règles de priorité, etc. Face à de tels comportements, les systèmes de confiance et réputation permettent une réponse adaptée dans de nombreux cas et sont ainsi un outil extrêmement important.

Par ailleurs, si les outils cryptographiques classiques sont très efficaces contre des attaquants externes, cela ne sera pas le cas dans de nombreuses situations face à des attaquants internes. En effet, ces attaquants pourront observer des informations variées lors

de leur participation aux différents protocoles collaboratifs dans le réseau. Par exemple, si deux nœuds échangent des messages chiffrés avec un protocole de routage classique, il pourront occulter le contenu des messages, mais ils devront bien informer les nœuds intermédiaires de la destination de leurs messages, et donc il sera difficile d'occulter qui communique avec qui face à un attaquant interne, même si on utilise un système de chiffrement. De la même façon, quand différents nœuds vont évaluer une fonction collaborativement (par exemple une fonction de majorité pour un vote, une moyenne pour une mesure globale, etc.) ils vont échanger des informations (leur vote, leur mesure) qu'ils ne pourront pas chiffrer de façon traditionnelle s'ils veulent qu'elles soient utilisées pour le calcul. Pour limiter l'intérêt des informations que ces attaquants vont pouvoir obtenir, il est possible d'utiliser des mécanismes de protection de la vie privée qui permettent par exemple de router des messages en protégeant l'identité des nœuds communiquant ou de réaliser des calculs distribués en occultant les valeurs des différents participants au calcul. Par exemple, l'utilisation des systèmes de confiance et de réputation dans des réseaux sans infrastructure tels que les réseaux ad hoc, nécessite une organisation collaborative des nœuds du réseau. Tout d'abord, les entités calculent localement leurs propres valeurs de confiance et agrègent par la suite leurs résultats. Si la politique de sécurité menée par l'application est orientée vers la protection de la vie privée de ses utilisateurs, il devient important de trouver des moyens sûrs et efficaces qui permettent l'évaluation de la fonction d'agrégation sans qu'aucun utilisateur ne divulgue ses valeurs de confiance.

Il existe deux manières distinctes d'assurer la protection de la vie privée dans un système de confiance. La première approche consiste à cacher la vraie identité des utilisateurs lors d'un échange de valeurs de confiance. Selon les objectifs tracés par l'application, on trouve dans la littérature plusieurs techniques permettant d'assurer les différentes propriétés d'anonymat [PH09]. Le mécanisme de base le plus utilisé consiste à affecter à chaque entité du système une pseudo-identité. L'inconvénient d'appliquer ces méthodes aux systèmes de confiance réside dans le fait qu'il est difficile d'agir sur les utilisateurs surveillés en ne connaissant que leur pseudo-identité (p. ex., afin de les inciter à un bon comportement, de les sanctionner, etc.). De plus, dans la plupart des solutions proposées, il existe toujours un moyen permettant aux utilisateurs de changer facilement leurs identités, ce qui peut engendrer de nouveaux problèmes tels que l'*attaque Sybille*.

La deuxième approche consiste à effectuer les calculs et les échanges des valeurs de confiance sans que les différentes parties ne divulguent leurs informations personnelles. Cette problématique est traitée dans le domaine de la sécurisation des calculs multiparties

(Secure Multi-Party Computations problem (SMPC)). Le problème de la sécurité des calculs multiparties est de permettre à différentes parties de réaliser des calculs, sur leurs données privées $(x_1, x_2, ..., x_n)$, de manière à ce que chacun connaisse son propre résultat final : $f_i(x_1, x_2, ..., x_n)$, mais sans qu'aucune partie ne puisse déduire les données privées des autres.

L'établissement des relations de confiance est souvent basé sur un opérateur de concaténation pour former par transitivité des chemins de confiance et sur un opérateur d'agrégation de ces chemins pour former la note de confiance globale. Dans la pratique, les deux opérateurs correspondent respectivement au produit arithmétique et à l'addition arithmétique. Autrement dit, les fonctions de calcul et de dérivation de la confiance sont essentiellement des variantes de l'équation 3.4. En calculant ces fonctions de manière à protéger les données privées (valeurs de confiance ou de réputation) des différents participants, on parvient à assurer la sécurité du système de confiance du point de vue de la protection de la vie privée.

On retrouve dans la littérature plusieurs travaux sur la sécurisation d'une somme multipartie ou d'un produit scalaire multipartie. Dans chaque proposition, on essaye de donner le schéma le mieux adapté à l'application traitée. On recense deux travaux très intéressants dans le cadre des applications de gestion de confiance et de réputation.

Dans [PRT04], les auteurs proposent un protocole permettant d'assurer la protection de la vie privée dans un système de réputation additif. Leur schéma s'appuie sur l'échange de secrets de Shamir [Sha79] et sur un chiffrement homomorphe. Leur protocole respecte le modèle semi-honnête et si on suppose que le nombre de parties est n, ils ont prouvé que leur schéma est résistant à une collusion constituée d'un nombre de parties pouvant aller jusqu'à $n - 1$. Malheureusement, leur schéma engendre un flux de communication considérable, évalué à $O(n^3)$ messages.

Dans [YTP07], les auteurs proposent un protocole permettant le calcul d'un produit scalaire multipartie, protégeant les données privées des participants et adapté aux systèmes de confiance plus complexes. Yao et *al.* se proposent de calculer le produit scalaire de deux vecteurs :

- Le premier vecteur représentant les valeurs des recommandations ;
- Le second vecteur représentant les valeurs de confiance en ces recommandations.

Dans leur article, Yao et *al.* ont basé leurs travaux sur ceux effectués auparavant dans le

cas de la sécurisation des produits scalaires à deux parties [AD01, GLLM04] afin de proposer un protocole efficace et sécurisé faisant intervenir plus de deux parties, c'est-à-dire que les valeurs des deux vecteurs peuvent appartenir à plusieurs entités. Les données privées de chacune des différentes parties sont gardées secrètes hormis le résultat final du produit scalaire qui, quant à lui, peut être connu par tous les intervenants. Leur schéma repose sur le modèle semi-honnête et est résistant à une collusion constituée au maximum de quatre parties. Leur schéma ne nécessite pas beaucoup de ressources matérielles et engendre une complexité en termes de communication estimée à $O(n)$ messages.

Nous avons envisagé, dans un premier temps, d'étendre le modèle décrit dans [PRT04] pour l'adapter au calcul multipartie d'un produit scalaire. Cependant, cette nouvelle version hérite des mêmes inconvénients que la version originelle à savoir un coût de communication trop élevé. Dans un second temps, nous avons amélioré le protocole proposé par Yao et *al.*. La solution que nous proposons est basée sur le protocole d'envoi superposé (DC-network). Grâce aux propriétés de sécurité de ce protocole, nous obtenons une solution dont la sécurité est prouvée dans un modèle semi-honnête et offrant une résistance optimale aux collusions. Ce schéma est indépendant du type d'application et peut être adapté à toute application nécessitant comme briques de base des produits scalaires distribués.

Dans la suite de ce chapitre, nous allons commencer par présenter dans la section 3.1 les systèmes de gestion de confiance et de réputation. Par la suite, nous introduirons dans le section 3.2 les problématiques des calculs multiparties ainsi que les différents modèles d'attaquant utilisés dans ce domaine. Dans la section 3.3, nous allons étudier le chiffrement homomorphe qui est une primitive cryptographique très utilisée dans la sécurité des protocoles de calcul multiparties, et l'envoi superposé qui va constituer avec le chiffrement homomorphe les deux briques de base de notre proposition. Après avoir présenté, dans la section 3.5, l'état de l'art des protocoles de protection des données privées dans un système de confiance et de réputation, nous présenterons notre proposition dans la section 3.6. La dernière section de ce chapitre sera consacrée aux conclusions et perspectives.

3.1 *Systèmes de gestion de confiance et de réputation*

Dans cette section, nous allons discuter des méthodes utilisées pour établir les relations de confiance entre les entités d'un réseau de communication et tout particulièrement dans le cas d'un réseau ad hoc. Après une courte introduction, nous donnerons quelques définitions de la notion de confiance telle qu'elle est habituellement utilisée dans la vie de tous les jours

et aussi telle qu'elle est utilisée en informatique. Ensuite, nous étudierons la manière avec laquelle la confiance est évaluée, propagée et utilisée dans les réseaux distribués tels que les réseaux ad hoc.

Les systèmes de gestion de confiance et de réputation tiennent une place très importante dans la littérature relative à la sécurité des réseaux ad hoc. Un des plus importants problèmes de sécurité des réseaux ad hoc réside essentiellement dans leur nature distribuée. Ces réseaux ad hoc reposent sur la collaboration des nœuds les composant afin de réaliser les fonctionnalités courantes du système. De ce fait, afin que la collaboration soit productive, il est nécessaire que la majorité des participants adoptent un comportement honnête. Pour cela, il est nécessaire de donner au réseau les moyens d'inciter les nœuds à collaborer et à adopter des comportements en adéquation avec les exigences des applications. Les systèmes de gestion de la confiance répondent à ce besoin.

Il existe trois avantages majeurs liés à l'utilisation d'un système d'évaluation de la confiance dans un réseau distribué. Premièrement, le fait de disposer d'une méthode d'évaluation de la confiance offre une incitation à un bon comportement des nœuds. Deuxièmement, l'évaluation de la confiance offre aux nœuds la possibilité de prédire le comportement futur des autres nœuds. Cette prédiction permet d'aider le nœud dans sa prise de décision. En d'autres termes, elle permet aux nœuds honnêtes d'éviter d'interagir avec les nœuds les moins fiables, ce qui réduit la participation des nœuds malveillants aux opérations du réseau. Troisièmement, le résultat du processus d'évaluation de la confiance peut être utilisé directement dans la détection des nœuds malfaisants et égoïstes dans le réseau et à la mise en place de sanctions.

3.1.1 Notions de la confiance

Qu'est ce que la confiance ?

Il faut tout d'abord noter qu'il n'existe pas une seule définition de la notion de confiance. Au lieu de cela, il apparaît une multitude de définitions qui varient selon la méthodologie utilisée pour observer le phénomène et selon le contexte dans lequel elle est employée. Quotidiennement, des mécanismes de confiance sont employés pour favoriser les relations sociales, amicales, familiales, etc. Certaines personnes sont prêtes à accepter un risque dans des situations où elles ont seulement une connaissance partielle de la réalité. Dès lors qu'il est nécessaire d'agir dans une situation indéterminée, de réagir aux actions entreprises par d'autres ou simplement de prendre une décision vis-à-vis d'un choix complexe, la confiance

permet d'avancer raisonnablement dans une voie à partir de connaissances partielles.

La manifestation de la confiance dans la vraie vie est très facile à percevoir, car on s'appuie constamment sur elle. Paradoxalement à ça, il est très difficile de la définir, dans la mesure où, elle se manifeste sous différentes formes. Dans la littérature aussi le terme de confiance peut prêter à confusion, car la confiance y est utilisée selon différentes significations [MC96].

En analysant les différents travaux de recherche traitant de la confiance, nous avons retenu les définitions suivantes qui répondent le plus aux besoins de l'informatique.

Selon Gambetta [Gam88], la confiance peut être interprétée comme étant la fiabilité de quelque chose ou de quelqu'un.

Définition 1. *(Confiance en terme de fiabilité) : La confiance est la probabilité subjective par laquelle un individu, A, prévoit qu'un autre individu, B, exécute une action donnée dont son bien-être dépend.*

Cette définition met en évidence deux notions essentielles : la notion de dépendance vis-à-vis d'une personne dite de confiance et la notion de valeur (probabilité) subjective (sentiment, croyance) qu'on associe à la capacité de quelqu'un ou quelque chose à réaliser une action. Cependant, le concept de confiance est plus complexe que ce que ne laisse supposer cette définition. Par exemple, Falcone et Castelfranchi [FC01] reconnaissent qu'il ne suffit pas d'avoir confiance, en termes de fiabilité, en une personne pour décider de lier une relation de dépendance avec elle. En effet, il arrive souvent que le risque en cas d'échec est tel que même si on a une grande confiance en une personne on hésite à entrer en relation avec elle. Une définition plus générale de la notion de confiance, qui inclut la notion de risque, est donnée dans [JIB07] :

Définition 2. *(Confiance basée sur la décision) : La confiance est la dépendance qu'une entité est prête à accepter vis-à-vis d'une chose ou d'une personne dans une situation donnée avec un sentiment de sécurité relative, même si des conséquences négatives sont possibles.*

Cette définition, tout en étant relativement vague, est très intéressante dans la mesure où elle est plus générale. Cette définition inclut implicitement et explicitement les ingrédients de base de la confiance. En plus des deux aspects de la confiance que nous avons abordés plus haut et qui sont "la dépendance en une personne dite de confiance" et "la fiabilité", on peut en faire ressortir deux autres aspects intéressants qui sont "l'utilité" et "la notion de risque". On va parler d'utilité positive si les résultats attendus sont positifs et d'utilité négative si les résultats attendus sont négatifs. La notion de risque apparaît quant à elle, par

exemple, lorsque les enjeux d'une transaction sont élevés et que la probabilité d'échec est non négligeable.

Qu'est ce que la réputation ?

Regardons maintenant de plus près la notion de réputation. La notion de réputation est étroitement liée à celle de la confiance, mais néanmoins, il existe des différences claires et importantes entre ces deux notions. Afin de faire ressortir ces différences, voyons la définition de la réputation proposée par le dictionnaire Larousse, qui est aussi celle reprise dans les nombreux travaux autour des réseaux sociaux :

Définition 3. *(Réputation) : Opinion favorable ou défavorable du public pour quelqu'un, quelque chose.*

D'après cette définition, on peut dire que la réputation d'un individu n'est que le fruit de l'agrégation des opinions de certaines personnes de sa communauté. La différence la plus importante peut être illustrée par cet état de fait : "on peut toujours avoir confiance en une personne malgré sa mauvaise réputation". Effectivement, on peut toujours aller au-delà de la réputation d'un individu et décider de lui faire confiance ou pas, et ceci, en donnant plus de poids à sa propre connaissance de l'individu. Ceci montre bien que la confiance est un phénomène subjectif qui est basé sur un certain nombre de critères et de preuves, et dont quelques-uns ont un poids plus important que les autres. L'expérience personnelle est souvent privilégiée par rapport aux recommandations, mais en l'absence d'informations personnelles, la confiance peut être basée sur les recommandations des autres. La réputation peut être alors considérée comme une mesure collective de la confiance (en termes de fiabilité) qui s'appuie sur les notations (évaluations) des autres membres d'une communauté. La confiance subjective d'un individu peut être aussi obtenue en combinant son expérience personnelle et les recommandations reçues.

Qu'est ce que la dérivation de la confiance par transitivité ?

Une notion très importante et qui est très utilisée dans l'établissement de la confiance, est la dérivation de la confiance par transitivité [JP05]. L'idée générale de cette notion est la suivante. Prenons trois entités Alice, Bob et Eric (voir figure 3.1). Si Alice fait confiance à Bob et que Bob fait confiance à Eric, et si Bob recommande Eric à Alice, alors Alice peut à son tour faire confiance à Eric en se basant sur les recommandations de Bob combinées avec sa propre connaissance d'Eric. C'est notamment la technique utilisée dans PGP [Zim95] pour construire une confiance envers l'identité d'une clé publique.

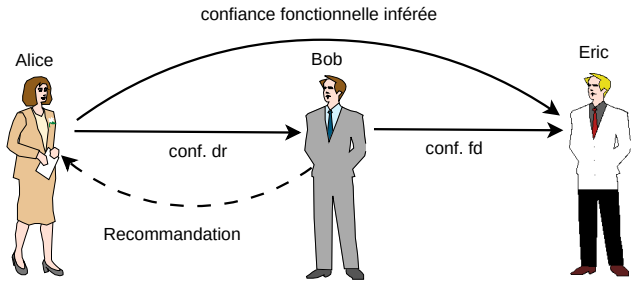

FIGURE 3.1 – Notion de confiance par transitivité

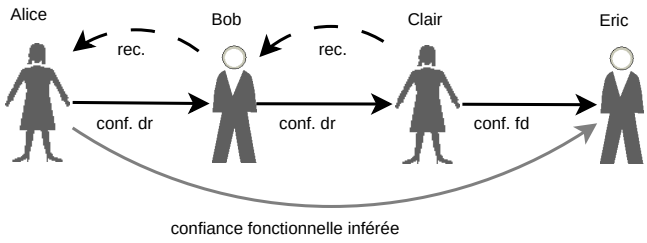

FIGURE 3.2 – Propagation de la confiance par transitivité

Exemple 1. *(Authentification) : Considérons la relation de confiance "A accepte l'authentification de E certifiée par B", qui est établie entre les entités A, B et E. Cette relation est la composition de deux relations de base qui résultent de deux processus distincts d'établissement de confiance ; c'est-à-dire "L'autorité B accepte les preuves de confiance relatives à l'authenticité de E", et "L'autorité A accepte toutes les authentifications délivrées par B". La première relation est établie par B en évaluant, hors ligne, les preuves de confiance présentées par E. B va exiger de E plusieurs pièces de preuves attestant de son identité. La deuxième relation, est établie par A, hors ligne, en évaluant les preuves de confiance attestant du domaine d'autorité de B.*

La confiance n'est pas toujours transitive. Par exemple, le fait qu' Alice fasse confiance à Bob pour garder ses enfants et que Bob fasse confiance à Eric comme étant un bon chirurgien esthétique, n'implique pas forcément qu'Alice peut faire confiance à Eric ni pour garder ses enfants ni comme chirurgien. Ceci est dû au fait que les actions évaluées par Bob et Alice sont différentes (les deux champs d'application de la confiance sont différents). Toutefois, sous certaines contraintes sémantiques [JP05], la confiance peut être transitive et ainsi elle peut être utilisée par les systèmes de calcul de confiance.

En théorie, il est important de distinguer entre les différentes classes de confiance. Premièrement, Il faut faire la distinction entre la confiance en les recommandations des autres et la confiance en la capacité d'une entité à réaliser une action, telle qu'elle a été définie plus haut (*confiance fonctionnelle*). Deuxièmement, il faut différencier la confiance directe de la confiance indirecte. Si on reprend l'exemple précédent. Si on suppose que Bob a prouvé à Alice, à maintes reprises, qu'il était un bon connaisseur du milieu de la chirurgie esthétique, la confiance que Alice accorde à Bob pour lui recommander un chirurgien esthétique est considérée comme étant directe. Par ailleurs, si on suppose aussi que Eric a prouvé à Bob, par le passé, qu'il était un bon chirurgien esthétique, la confiance fonctionnelle de Bob en Eric est aussi considérée comme étant directe. Grâce aux conseils de Bob, Alice peut aussi faire confiance à Eric. Cependant, cette confiance fonctionnelle n'est pas directe, car Alice n'a pas eu l'occasion d'observer le chirurgien Eric à l'oeuvre. Elle est donc considérée comme une confiance indirecte inférée par transitivité (voir figure 3.2).

Les systèmes implémentés dans la pratique prennent rarement en compte toutes ces subtilités et s'appuient généralement sur des méthodes simples. Ils ne font généralement pas de distinction entre : la confiance fonctionnelle et la confiance en les recommandations des autres ; la confiance directe et indirecte ; et ne tiennent pas compte du domaine d'application. Ceci à cause du fait qu'il est difficile d'obtenir suffisamment d'informations afin de faire la différence entre les différentes sémantiques de la confiance, et aussi par ce que ceci exigerait des modèles de confiance trop complexes.

La confiance et la réputation en informatique

Même si l'utilisation de la confiance dans la vie de tous les jours semble être une tâche assez simple et intuitive, sa définition et son utilisation dans le monde virtuel sont un peu plus complexes. Effectivement, tous les critères et tous les indicateurs qu'on utilise habituellement pour établir les relations de confiance entre individus, ne sont pas exploitables en informatique. Par conséquent, la première difficulté dans la mise en place d'un système de confiance et de réputation est d'identifier les nouveaux critères et preuves de confiance à prendre en considération dans l'établissement des relations de confiance et de trouver de nouveaux indicateurs spécifiques à chaque application. En effet, les critères et preuves de confiance qui seront pris en compte dans une application de vente en ligne ne seront pas les mêmes que ceux qui seront pris en compte dans un protocole de routage ou bien dans un protocole de partage de fichiers dans un réseau pair-à-pair.

En outre, les communications et le partage d'informations sont relativement difficiles et souvent contraints dans la vraie vie, alors qu'en informatique cette tâche est facile. Par exemple, grâce à Internet, il peut être mis en place des systèmes très efficaces permettant des échanges d'informations à grande échelle.

À partir de ces deux constatations, la conception d'un système de confiance et de réputation doit être axée sur ces deux points :

- Trouver des substituts aux critères et aux indicateurs utilisés dans la vraie vie et identifier, pour chaque application, de nouveaux éléments d'information adéquats pour construire des métriques afin de mesurer la confiance et la réputation ;
- Profiter des avantages de l'informatique et des télécommunications afin de construire des systèmes de collecte et de distribution d'informations.

En d'autres termes, les chercheurs doivent répondre à toutes ces questions fondamentales : quelles informations sont les plus appropriées pour mesurer la confiance et la réputation dans une application donnée ? De quelle manière ces informations sont capturées et collectées ? Quelles sont les meilleures méthodes à utiliser afin de concevoir de tels systèmes du point de vue théorique et pratique ? Ces méthodes sont elles résistantes aux attaques ?, etc..

Afin d'éviter toute ambiguïté, nous allons essayer de donner une définition de la confiance et de la réputation telles qu'elles sont utilisées dans les systèmes de gestion de la confiance, en synthétisant toutes les notions qu'on vient de voir jusqu'à maintenant.

Définition 4. *La confiance est une relation unidirectionnelle entre deux entités participant à un protocole. Cette relation est basée sur l'évaluation des informations reçues ou sur l'évaluation des interactions passées entre ces deux entités dans le cadre de l'exécution du protocole. La relation de confiance dépend du domaine d'application dans lequel elle est utilisée. L'application (le protocole) détermine la sémantique de la confiance et définit la façon dont la confiance est calculée et partagée.*

D'après cette définition, la confiance est établie entre deux entités par rapport à une action spécifique. Une entité fait confiance à une autre entité selon sa capacité à réaliser une certaine action dépendant d'une application donnée telle que "être authentique" dans le cas des systèmes de gestion de clés, "le paquet a-t-il été routé correctement" dans le cas des protocoles de routage, "un capteur a-t-il correctement récolté les données attendues" dans le cas d'un réseau de capteurs, etc..

3.1.2 Architectures

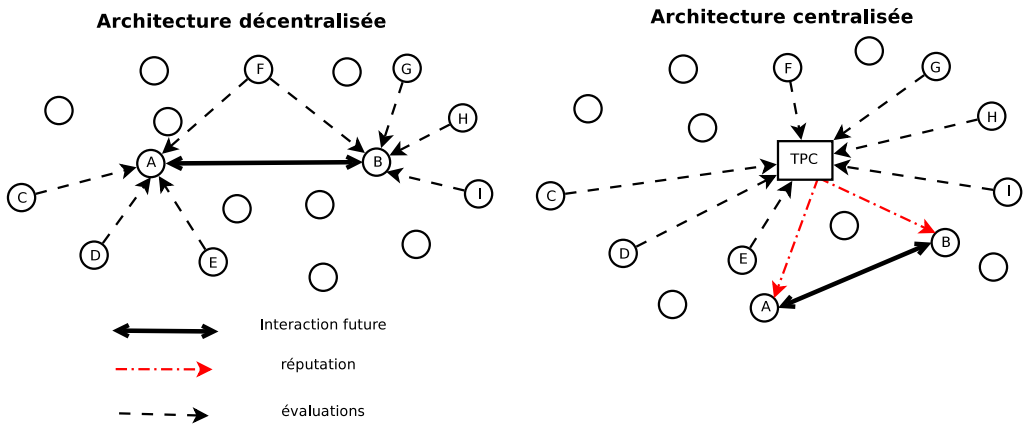

FIGURE 3.3 – Architectures des systèmes de gestion de confiance et de réputation

L'architecture d'un réseau détermine la manière dont les utilisateurs du système s'échangent leurs évaluations. Les deux principaux types d'architecture existants sont les architectures centralisées et les architectures décentralisées (cf. figure 3.3).

Les systèmes de confiance et de réputation centralisés s'appuient sur une tierce partie de confiance qui s'occupe d'une part de collecter les évaluations des participants et d'autre part d'inférer, sur la base des évaluations collectées, la réputation de chaque participant au système. Les notes de réputation sont ensuite rendues publiques.

Il existe des systèmes où les architectures de confiance et de réputation décentralisées sont plus appropriées qu'une architecture centralisée reposant sur une unique autorité de confiance. C'est notamment le cas des réseaux ad hoc. Dans ces architectures, les nœuds s'auto-évaluent, s'échangent leurs évaluations pair-à-pair, sauvegardent les évaluations reçues et dérivent localement les notes de confiance.

3.1.3 Métriques d'évaluation de la confiance

Interpréter la confiance comme une mesure d'évaluation de la fiabilité d'une entité a permis l'émergence de nombreuses métriques.

La confiance est évaluée suivant différentes approches. Certains modèles emploient une description linguistique de la relation de confiance [Zim95, ARH97, HMM+00]. Les systèmes offrant une telle expression de la confiance sont beaucoup plus expressifs que ceux utilisant une valeur de probabilité. Cependant, cette expression linguistique rend très difficile la

réalisation de calculs autour des évaluations de confiance. Ainsi, dans de nombreux cas, les systèmes de confiance utilisent des méthodes probabilistes. Ces approches sont très intéressantes, car on peut leur appliquer directement la multitude de méthodes probabilistes existantes. Ces modèles offrent une variété de méthodes de dérivation. On peut trouver de simples modèles basés sur des calculs de probabilité basiques : par exemple dans [CJI02], leur système de réputation est basé sur la loi de probabilité binomiale (Béta) qui est souvent utilisée pour représenter la distribution de probabilité a posteriori d'un événement binaire. De manière plus précise, soit une interaction entre deux entités et soit (r_1,r_2) un couple de deux valeurs continues qui pourraient représenter respectivement le degré de satisfaction et de non-satisfaction du point de vue de l'entité qui évalue, et ce, après que la transaction ait eu lieu. La probabilité que la prochaine interaction soit positive est donnée par la distribution de probabilité p dont l'espérance $E(p) = \frac{r_1}{r_1+r_2}$. D'autres méthodes plus générales que cette dernière, basées sur des lois de probabilité multinomiales, ont été proposées. La méthode la plus connue est celle basée sur les systèmes bayésiens [MMA+01, MMH02, WJI04].

3.1.4 Opérateurs de propagation de la confiance

Comme nous l'avons définie précédemment, la confiance accordée à une entité peut être évaluée suivant des observations directes en utilisant différentes métriques d'évaluation de la confiance, ou bien indirectement à travers plusieurs autres entités. Le problème qui se pose souvent dans de tels modèles est de trouver le moyen efficace pour propager la confiance entre les différentes entités d'un système. À cet effet plusieurs opérateurs de propagation, de combinaison et d'agrégation des notes de confiance ont été définis.

On peut modéliser les différentes relations de confiance entre les entités d'un système suivant un graphe orienté et valué G où les sommets du graphe représentent les différentes entités du système et où les arcs représentent les relations de confiance entre elles. Il existe un arc valué et orienté entre deux entités X et Y , si X connaît Y et le poids de l'arc les reliant représente la note de confiance $C(X, Y)$ attribué par X à Y. La distance entre X et Y dépend du chemin utilisé pour calculer cette note de confiance. Ce chemin est appelé le chemin d'établissement de confiance entre X et Y.

On retrouve dans la littérature plusieurs opérateurs de transfert et de propagation de la confiance [HWS09]. Cependant, les deux opérateurs les plus usités pour transférer la confiance dans un système de réputation sont l'opérateur de concaténation noté $\otimes$ et l'opérateur d'agrégation noté $\oplus$.

L'opérateur de concaténation permet de modérer une note de confiance calculée par transitivité. En d'autres termes, lorsqu'un nœud X établit une relation de confiance avec

un nœud Y à travers les recommandations d'une tierce partie Z, la note de confiance entre les nœuds X et Y ne devrait pas dépasser d'une part, la note de confiance entre le nœud X et la tierce partie et d'autre part, la valeur de confiance entre la tierce partie et le noeud Y. Pour cela, certains modèles incluent le degré de confiance en les recommandations d'une tierce partie de confiance $CR(X, Z)$. De manière plus formelle :

$$C(X, Y) = CR(X, Z) \otimes C(Z, Y) \leq min(CRX, Z), C(Z, Y)). \tag{3.1}$$

Dans le cas où Y ne connaît pas directement Z. La formule donnant la note de confiance est la suivante :

$$C(X, Y) = CR(X, Z_1) \left(\bigotimes_{i=1}^{k} CR(Z_i, Z_{i+1}) \right) C(Z_{k+1}, Y). \tag{3.2}$$

Afin d'augmenter la précision des évaluations, il est souvent nécessaire d'obtenir des recommandations de la part de multiples sources et suivant des chemins de confiance différents. L'opérateur d'agrégation permet justement de combiner ces recommandations. De manière plus formelle, si on combine les deux opérateurs ensemble, on obtient la formule globale d'établissement de confiance suivante :

$$C(X, Y) = \bigoplus_{i=1}^{n} \left(CR(X, Z_{i,1}) \left(\bigotimes_{j=1}^{k} CR(Z_{i,j}, Z_{i,j+1}) \right) C(Z_{i,k+1}, Y) \right). \tag{3.3}$$

Dans la pratique, les opérateurs qui correspondent intuitivement aux opérateurs de concaténation et d'agrégation sont respectivement les opérateurs de multiplication et d'addition. La formule de calcul de la note de confiance globale 3.3 devient donc :

$$C(X, Y) = 1/n \sum_{i=1}^{n} \left(CR(X, Z_{i,1}) \left(\prod_{j=1}^{k} CR(Z_{i,j}, Z_{i,j+1}) \right) C(Z_{i,k+1}, Y) \right). \tag{3.4}$$

3.1.5 Systèmes de gestion de confiance dans les réseaux ad hoc

Comme dit en début de ce chapitre, les réseaux ad hoc sont un exemple type de systèmes qui nécessitent la coopération de tous leurs participants pour bien fonctionner. Toute déviation d'un participant du comportement autorisé par la politique mise en place dans le système influe négativement sur la bonne marche du réseau. Prenons l'exemple de la fonction de routage dans les réseaux ad hoc ou dans les réseaux de capteurs. Comme il n'existe pas d'infrastructure de routage, un nœud est libre sur sa façon de gérer les paquets qu'il reçoit. Il peut décider de se comporter conformément aux règles de routage (faire suivre

convenablement ces paquets) ou bien d'adopter un comportement contraire aux règles définies dans le réseau. En effet, le nœud peut décider de ne pas router les messages de ses voisins (le nœud adopte un comportement égoïste) ce qui est à l'opposé de l'esprit de coopération des réseaux ad hoc. Le nœud peut aller encore plus loin et remettre en cause l'intégrité des données qu'il reçoit (modifier les paquets reçus en particulier les paquets de contrôle qui sont dans la majorité des cas non chiffrés). Le nœud peut aussi injecter de fausses informations de routage qui auront pour conséquence de corrompre les tables de routage de certains nœuds du réseau et ainsi fausser la fonction de routage.

Une solution à ces problèmes est de mettre en place un système de surveillance du comportement des nœuds. En effet, en permettant à chaque nœud d'évaluer le comportement de ses voisins à un saut et en partageant ces évaluations avec les autres nœuds, chaque nœud du réseau pourra avoir une opinion assez précise de ses voisins. Les systèmes de réputation appliqués aux réseaux ad hoc fonctionnent, de manière générale, de la même façon que nous les avons définis un peu plus haut. Ils utilisent trois mécanismes distincts :

- évaluations effectuées localement par chaque nœud en observant le comportement de leurs voisins ;
- un mécanisme de partage des évaluations obtenues localement entre les nœuds du réseau ;
- un mécanisme de sanction ou d'isolation des comportements anormaux.

Les évaluations portent essentiellement sur les interactions réalisées dans le cadre du routage (un nœud relaie-t-il vraiment les paquets qu'un voisin lui envoi ? Est ce qu'un nœud n'altère pas les données qui lui sont envoyées ?...). Le problème qui est posé alors, est de trouver un moyen permettant à un nœud de répondre à ces questions. En effet, une des caractéristiques des réseaux sans fil ad hoc est l'accès partagé au média de communication, tous les nœuds peuvent écouter les informations transmises par leurs voisins à un saut, même si ces dernières ne leur sont pas destinées. Il s'agit du mode de fonctionnement *promiscuous*. A l'aide de ce mécanisme chaque nœud sera en mesure de voir, par exemple, si un des voisins qu'il surveille relaie réellement les messages qu'il reçoit.

Même si cette technique est très utilisée, elle présente cependant quelques limitations [MGLBoo]. En plus des problèmes techniques liés à la mise en place de ce mode de fonctionnement de la carte réseau (interfaces réseaux ou pilotes ne gérant pas ce mode,

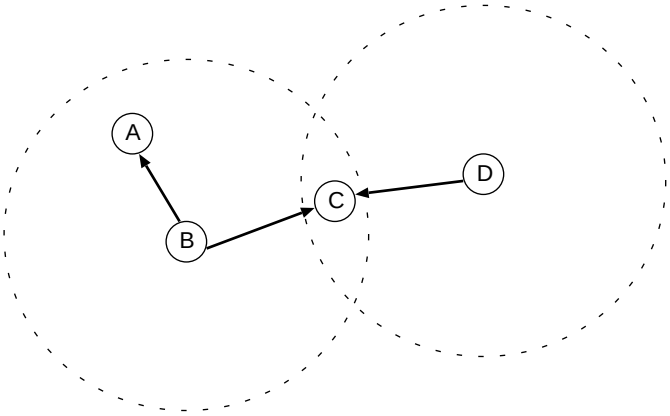

FIGURE 3.4 – Inconvénients de la surveillance basée sur le mode promiscuous

augmentation de la consommation d'énergie due au fait que les nœuds gèrent aussi les paquets destinés à leurs voisins), cette méthode génère un certain nombre de faux positifs et de faux négatifs, puisque le nœud utilisant ce mode de surveillance n'a pas une vue précise sur des éventuels succès ou échecs des transmissions de ses voisins. Pour illustrer ceci, voici trois exemples de situations ou le nœud *C*, après avoir envoyé un paquet de données au nœud *B* à destination de *A*, surveille *B* quant à son intention de retransmettre ce paquet à sa destination (figure 3.4).

Exemple 2. *On peut remarquer sur la figure 3.4 que le nœud C est à l'intérieur de la portée radio du nœud B, ce qui signifie qu'il peut utiliser le mode promiscuous pour vérifier si le nœud B retransmet comme prévu le paquet à A. Maintenant, supposons que D transmet un paquet au même moment que B route le paquet de C vers A. Comme C est aussi à l'intérieur du champ de transmission de D, il ne pourra pas dire si le nœud B a réellement retransmis le paquet à cause de la collision entre les deux transmissions de B et D. De ce fait, le nœud C pourrait accuser le nœud B d'être un nœud égoïste alors que ce n'est pas le cas (faux positif).*

Exemple 3. *Le nœud B provoque une collision (il transmet le paquet de C au même moment que A ou un autre voisin transmet un paquet) que le nœud C pourrait ne pas voir s'il n'est pas dans le champ de transmission de A (ce qui est le cas sur la figure 3.4). Si le nœud B ne retransmet pas le paquet, le nœud C ne pourra pas détecter le comportement malveillant de B (faux négatif)*

Exemple 4. *Un autre cas de faux négatif. C'est le cas où le nœud A est un peu plus loin que le nœud C de B, le nœud B peut tromper le nœud C en retransmettant le paquet avec un faible signal qui atteint C mais non A.*

Un autre problème de cette technique est lié aux changements fréquents de la topologie des réseaux ad hoc (mobilité des nœuds). En effet, un nœud utilisant le mode promiscuous ne pourra pas toujours observer le trafic de tous les paquets reçus ou expédiés par ses voisins à cause des coupures fréquentes et temporaires de liens avec ces derniers.

Malgré ces imperfections, l'utilisation des mécanismes de surveillance basés sur le mode promiscuous est un moyen utile permettant d'évaluer le comportement d'un nœud.

Les premiers protocoles proposés utilisant le mode promiscuous sont ceux de Marti et *al.* [MGLBoo]. Dans leurs travaux, les auteurs proposent un protocole basé sur le protocole de routage DSR. Leur schéma utilise deux composants : le chien de garde (Watchdog) et l'évaluateur de chemins (Pathrater). Le premier composant cité intervient au niveau de chaque nœud afin de contrôler que le nœud suivant sur le chemin procède bien à la retransmission des paquets vers le nœud suivant. Lorsqu'une action observée ne correspond pas au résultat attendu, le nœud observateur comptabilise un échec de retransmission. Dès que le nombre d'échecs comptabilisé par un nœud observateur dépasse le seuil fixé par le système, une alerte est remontée au Pathrater. Le Pathrater est responsable du calcul des évaluations et de la sélection des chemins les plus fiables en évitant les nœuds les moins coopératifs.

D'autres protocoles, beaucoup plus élaborés, ont été proposés. Bouchegger et Le Boudec ont proposé un système de renforcement de la coopération distribué et collaboratif dénommé CONFIDANT (COoperation of Nodes, Fairness In Dynamic Ad-hoc NeTworks) [BLBo2] appliqué au protocole DSR. L'objectif de CONFIDANT est d'exclure du réseau tout nœud égoïste, que ce soit au niveau du processus d'acheminement des données ou bien au niveau du processus de découverte du voisinage. On retrouve au niveau de chaque nœud quatre composants interagissant entre eux :

1. un moniteur ;
2. un système de réputation ;
3. un gestionnaire de confiance ;
4. un gestionnaire de chemins.

Le moniteur de chaque nœud évalue, sur la base de ses observations, le comportement des nœuds à l'égard de la fonction de routage. Une fois que le moniteur détecte un évènement suspicieux, il transmet l'information au système de réputation. Ce dernier maintient à jour les valeurs de réputation (évaluations) de chaque nœud observé. Afin de

remédier à l'imprécision des mécanismes de détection et d'avoir un système qui converge plus rapidement, CONFIDANT utilise le principe d'échange de recommandations entre les nœuds du réseau. Ce rôle est joué par le gestionnaire de confiance qui s'occupe d'une part, de la décision à prendre quant au partage des valeurs de réputation et d'autre part, de la façon d'agréger ensemble les recommandations reçues. CONFIDANT ne tient compte que des recommandations négatives. Finalement, le gestionnaire de chemins est utilisé dans le but de sélectionner les chemins optimaux en termes de fiabilité des nœuds qui les composent et peut aussi décider de ne pas router les paquets des nœuds ayant des notes de réputation globales en dessous du seuil toléré par le système.

Un autre protocole semblable en de nombreux points à CONFIDANT a été proposé par Michiardi et Molva [MMo2]. Le protocole s'appelle CORE (COllaboratif REputation mechanism) et il est basé lui aussi sur le protocole de routage DSR. La différence essentielle avec CONFIDANT réside dans la manière dont CORE calcule les valeurs de réputation. Dans CORE, les notes de confiance et de réputation globales sont calculées comme étant la combinaison de :

- l'évaluation subjective, calculée à partir des observations du nœud ;
- les évaluations indirectes (recommandations) positives, reçues des autres nœuds ;
- poids associés à chaque tâche spécifique.

Le mécanisme de pénalisation du système fonctionne de façon progressive, en excluant des activités du réseau, les nœuds ayant des notes de réputation et de confiance en dessous d'un seuil. Un mécanisme de rédemption offre la possibilité aux nœuds de réintégrer progressivement le réseau en coopérant de nouveau dans les opérations de routage.

Pour conclure notre étude sur les systèmes de confiance pour réseaux ad hoc, nous pouvons remarquer que les propositions que nous avons décrites jusqu'ici se basent exclusivement sur le comportement des nœuds. La limite de ce type de systèmes est qu'ils ne prennent pas en considération le contenu des paquets échangés entre les nœuds du réseau. Ainsi, un nœud peut avoir une bonne réputation, car il a fait preuve d'un bon comportement (il a par exemple routé convenablement les messages de ses voisins), mais les informations qu'il envoie peuvent être fausses (en particulier, celles relatives au routage). Dans ce contexte, nous pouvons citer comme exemple ces deux travaux très intéressants [WLMG05, CCBNR07].

3.2 *Sécurité des protocoles de calcul multiparties*

Avec la multiplication des applications utilisées dans les réseaux informatiques, la quantité de données collectées et traitées devient de plus en plus importante. Ces réseaux offrent l'opportunité à plusieurs utilisateurs de réaliser des calculs en coopérant et en mettant à disposition des uns et des autres leurs données privées. La plupart des données partagées sont des informations privatives qui reflètent soit les activités journalières des participants (p. ex., leurs itinéraires de voyage, leurs habitudes d'achat, etc.) s'il s'agit d'utilisateurs civils, soit d'informations commerciales, financières, militaires ou bien politiques dans le cas de calculs coopératifs entre organisations institutionnelles ou commerciales. De ce fait, il est important d'avoir des moyens permettant d'assurer la confidentialité de toute cette masse d'informations échangées, tout en permettant la bonne exécution de ces calculs. Ces problématiques sont englobées dans un axe de recherche appelé les calculs multiparties sécuritaires (aussi appelés protocoles de calcul multiparties sécurisés, protocoles de calcul multiparties préservant la vie privée, ou bien en anglais "Secure Multi-party Computations" (SMPC))

Les protocoles de calcul multiparties sécuritaires utilisent le plus souvent des techniques cryptographiques qui permettent de réaliser des calculs sur des données de différentes parties de manière à ce que chaque partie ne connaisse que ses propres informations ainsi que le résultat final du calcul. Dans un protocole de calcul multipartie, nous avons plusieurs individus (parties) $P_1, P_2, \cdots, P_m$, ayant chacun une donnée privée x_i (pour $1 \leq i \leq m$), qui désirent calculer la valeur d'une fonction publique f au point $(x_1, \cdots, x_m)$ (voir 3.5).

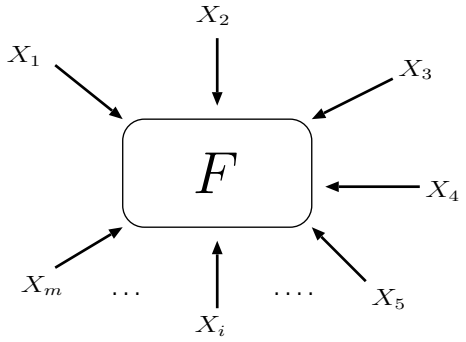

FIGURE 3.5 – Calcul multipartie

De manière plus générale, soit f une fonction calculable qui prend en entrée m éléments et produit m éléments en sortie. Sécuriser un calcul multipartie revient à trouver des techniques qui permettent à m parties $(P_1, P_2, \cdots, P_m)$, où chaque partie P_i connaît la valeur de la donnée x_i, de calculer conjointement $f(x_1, x_2, \cdots, x_m) = (y_1, y_2, \cdots, y_m)$ de manière à ce que chaque partie P_i ne puisse connaître que y_i. Ceci devrait être le cas même dans un scénario où un adversaire ayant une capacité de calcul suffisante tenterait de corrompre un sous-ensemble des m parties.

Sécuriser un protocole multipartie qui calcule une fonction f donnée revient à trouver des solutions contre des comportements anormaux (comportements ne respectant pas les règles spécifiées par le protocole) de certaines parties. En effet, l'objectif d'un attaquant externe contrôlant un sous-ensemble de parties lors de l'exécution du protocole est soit d'arriver à extraire des informations privées, soit de fausser les résultats du calcul. On peut donc distinguer deux types de besoins en termes de sécurité dans le cadre des protocoles multiparties : la protection des données privées et l'exactitude des résultats. L'exigence en termes de protection de la vie privée stipule qu'aucune information ne devrait être déduite au-delà de ce qui est absolument nécessaire (l'attaquant ne doit connaître que les données privées et les résultats des parties qu'il a pu corrompre et aucune autre information supplémentaire). L'exigence en termes d'exactitude des résultats stipule, quant à elle, que chaque partie devrait recevoir le résultat exact tel qu'il est défini par la fonction f.

Le problème de la sécurité des calculs multiparties englobe plusieurs domaines allant de simples protocoles de génération aléatoire à des protocoles plus complexes (data mining, private information retrieval, calculs géométriques, analyses statistiques, vote électronique, vente aux enchères en ligne, etc.). Par exemple, l'exigence en termes de protection de la vie privée dans un protocole de vote électronique assure qu'aucune partie ne doit connaître une quelconque information sur les votes des autres parties. L'exigence en termes d'exactitude des résultats permet quant à elle de s'assurer qu'aucune collusion de parties n'a la capacité d'influencer le résultat des votes au-delà de leur capacité à voter pour leur candidat préféré. De la même manière, dans un protocole de vente aux enchères, la première exigence assure que seul le gagnant de l'enchère doit être connu ; la seconde assure que c'est celui qui fait l'enchère la plus élevée qui gagne.

Historiquement, le problème de la sécurisation des calculs multiparties a été introduit par les travaux de Yao dans [Yao82] où il proposait une solution au fameux problème des deux

millionnaires (le problème était de connaître le plus riche des deux millionnaires sans que personne ne sache la valeur exacte des deux fortunes). Le problème des deux millionnaires de Yao [Yao82] est utile dans le contexte de l'exploration de données (data mining), dans le contexte des applications de commerce électronique (e-commerce) et tout particulièrement dans le cadre des ventes aux enchères et des appels d'offres sur Internet. Par la suite, d'autres travaux ont été menés dans ce sens. Plusieurs études théoriques dans ce domaine ont été réalisées, une des études les plus importantes étant celle faite par Goldreich [Gol05]. Dans cette étude, Goldreich affirme que tout problème SMPC peut être résolu en s'appuyant sur les protocoles d'évaluation de circuits.

Cependant, ces études restent très théoriques en raison des nombreuses contraintes liées à l'efficacité des solutions proposées. Par conséquent, il est important d'avoir des solutions adaptées à chaque type de problème et qui soient réalisables en pratique. Les recherches initiales autour de solutions spécifiques aux problèmes-types sont réalisées avec l'hypothèse d'un modèle de sécurité idéal. Cependant, en dépit du fait que ces solutions soient beaucoup moins coûteuses que les solutions génériques, elles restent, néanmoins, lentes lors de leur mise en œuvre.

3.2.1 Objectifs de sécurité

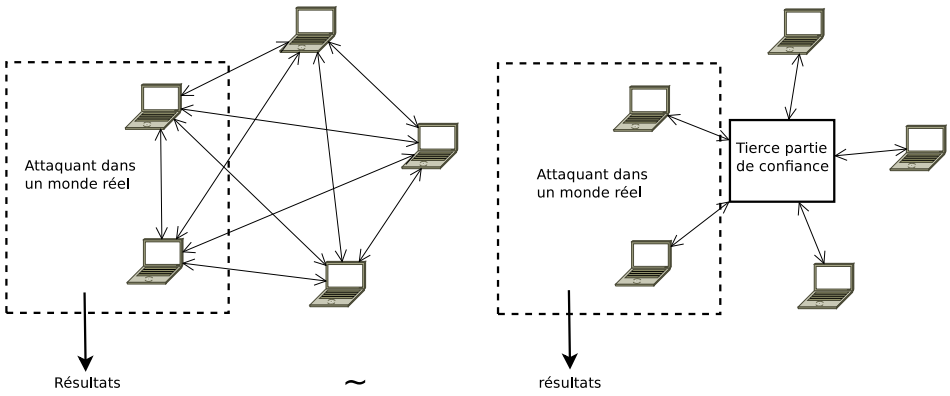

FIGURE 3.6 – Objectif de sécurité idéal d'un protocole de calcul multipartie

Dans le cadre de la sécurisation des protocoles multiparties, on dit d'un protocole multipartie qu'il est sécurisé contre un adversaire contrôlant un sous-ensemble de parties et opérant lors de son exécution, s'il satisfait certaines exigences de sécurité.

Protection des données privées : Aucune partie ne doit apprendre quelque chose de plus que le résultat de f prévu. Autrement dit, la seule information qui doit être connue à propos des entrées des autres parties est celle qui peut être dérivée du résultat de f. Par exemple, dans une vente aux enchères où la seule offre qui est révélée est celle du plus offrant, il est clairement possible d'en déduire que toutes les autres offres étaient plus faibles que l'enchère gagnante. Toutefois, cela devrait être la seule information qui puisse être révélée sur les autres offres.

Exactitude des résultats : Chaque partie doit avoir la garantie que le résultat qu'elle reçoit est correct. Pour continuer avec l'exemple d'une vente aux enchères, la partie qui fait l'offre la plus élevée doit être celle qui remporte l'enchère, et aucune partie, y compris le commissaire-priseur, ne peut modifier cela.

Indépendance des entrées : les parties corrompues doivent choisir leurs entrées indépendamment des entrées des autres parties. En reprenant l'exemple de la vente aux enchères, les parties doivent pouvoir fixer leurs offres de manière indépendante des autres. Il faut noter que la propriété de protection des données privées n'implique pas forcément cette propriété. Par exemple, il est possible de générer une enchère plus élevée qu'une autre sans connaître la valeur de cette dernière. Cette attaque peut être menée contre certains chiffrements tels que le chiffrement homomorphe (p. ex., soit le chiffrement homomorphe de 200, il est possible de calculer le chiffrement de $200 + x$, en ne connaissant que le chiffré de 200).

Garantir la livraison des résultats de f : les parties corrompues ne devraient pas être en mesure d'empêcher les parties honnêtes de recevoir leurs résultats. En d'autres termes, l'adversaire ne doit pas être en mesure de perturber le calcul en effectuant un déni de service.

Équité dans la livraison des résultats : les parties corrompues ne devraient recevoir leurs résultats que si et seulement si les parties honnêtes reçoivent également les leurs.

Une alternative à cette définition est basée sur deux modèles d'abstraction (cf. figure 3.6). Un modèle réel décrivant l'environnement et les conditions réelles d'exécution d'un protocole multipartie et un modèle idéal (aussi appelé "monde idéal") où les calculs sont supposés être réalisés par une tierce partie de confiance imaginaire qui est censée être incorruptible.

Dans un modèle d'exécution idéal, les parties envoient leurs données privées à la tierce partie de confiance, qui calcule la fonction désirée et renvoie à chaque partie son résultat attendu. Puisque les seules actions effectuées par les parties sont limitées à l'envoi de leurs données à la TPC, un adversaire n'aura comme possibilité d'attaque que de choisir les données des différentes parties sous son contrôle. Il faut noter que la plupart des propriétés de sécurité décrites ci-dessus sont vérifiées dans un modèle idéal. Par exemple, la protection des données privées est vérifiée, car le seul message reçu par une partie donnée est son résultat. De même, la propriété relative à l'exactitude des résultats est aussi vérifiée sachant que la TPC est incorruptible.

Le but de cette approche est de comparer les conséquences d'une attaque menée par un adversaire dans le contexte d'une exécution réelle du protocole avec les conséquences d'une attaque menée par un adversaire dans le cadre de l'exécution du même protocole dans des conditions idéales. Si tout ce que l'adversaire peut obtenir dans le cadre d'une configuration réelle d'exécution du protocole peut aussi être obtenu dans une configuration idéale (avec utilisation d'une tierce partie de confiance), il est dit du protocole "qu'il émule une configuration idéale d'exécution" (c.-à-d., qu'il émule une tierce partie de confiance). Puisque le modèle idéal est supposé être sécuritaire, on peut de ce fait affirmer que le protocole est sécuritaire.

Cette approche englobe de manière implicite toutes les exigences de sécurité que nous avons listées plus haut et peut s'étendre même au-delà. Par exemple, la protection des données privées résulte du fait que les résultats obtenus par l'attaquant sont les mêmes dans les deux modèles d'exécution. Sachant que dans le contexte d'une exécution idéale, l'attaquant ne connaît que les données des parties qu'il a corrompues, la même règle doit s'appliquer dans le cas d'une exécution réelle. La fiabilité des résultats de calcul résulte du fait que les résultats de calcul des parties honnêtes doivent être les mêmes dans les deux modèles, ainsi que du fait que dans le cas d'une exécution idéale du protocole, chaque partie obtient ses résultats de calcul tels qu'ils sont calculés par une tierce partie de confiance supposée être incorruptible et dont les calculs sont censés être fiables. En ce qui concerne l'indépendance des données des différentes parties en entrées du protocole, dans une exécution idéale, toutes les données privées sont envoyées à la TPC avant qu'un résultat soit émis par cette dernière. Par conséquent, les parties corrompues ne peuvent rien savoir sur les entrées des parties honnêtes en même temps qu'elles envoient leurs données.

Il est possible d'étendre ou de réduire la définition de sécurité présentée ci-dessus suivant

plusieurs directions et selon différents critères tels que la capacité de calcul de l'attaquant, le type de comportement adopté par l'attaquant, la stratégie utilisée pour contrôler les parties, etc.. En plus des paramètres qui concernent l'environnement d'exécution du protocole :

- l'existence d'une infrastructure à clés publiques où chaque partie partage une information secrète avec les autres parties,
- l'existence d'un canal sécurisé entre toutes les parties assurant la confidentialité et la fiabilité des messages échangés entre les parties honnêtes,
- l'existence d'un canal de diffusion direct permettant d'atteindre toutes les parties en même temps,

on peut aussi s'intéresser aux capacités et au comportement de l'adversaire. À cet effet, plusieurs paramètres sont souvent pris en considération dans la définition d'un modèle de sécurité :

- le nombre de parties sous le contrôle de l'adversaire,
- la capacité de calcul de l'adversaire,
- la façon dont l'attaquant externe a procédé pour sélectionner l'ensemble des parties qu'il contrôle. Le modèle de base et le plus simple est un ensemble déterminé avant le début d'exécution du protocole. Ce modèle est appelé le *modèle non adaptatif* qui est l'opposé du modèle *adaptatif* dans lequel l'adversaire peut sélectionner les parties qu'il va contrôler de manière adaptative durant l'exécution du protocole et selon les informations qu'il aura récoltées,
- la stratégie adoptée par les parties malhonnêtes pour attaquer le protocole. Les parties contrôlées par l'adversaire peuvent utiliser des étapes actives pour dysfonctionner l'exécution du protocole (p. ex., elles envoient des messages incohérents avec ceux spécifiés par le protocole), ou bien elles peuvent tout simplement tenter de récolter, de manière passive, un maximum d'informations sur les parties honnêtes et les partager entre elles. Il est donné à ce dernier comportement de l'adversaire différentes appellations, telles que le modèle *semi-honnête*, le modèle *honnête mais curieux* ou bien le modèle passif. Ce modèle représente une restriction du premier comportement de l'adversaire qui est quant à lui appelé le modèle *malveillant*.

3.2.1.1 Modèle d'attaquant semi-honnête

De manière générale, une partie semi-honnête est celle qui suit convenablement le protocole, mais qui peut aussi garder un enregistrement de tous ses calculs intermédiaires. On considère qu'elle sauvegarde tous les aléas générés en interne (p. ex., les masques et les clés secrètes) ainsi que tous les messages qu'elle reçoit des autres parties. C'est le modèle qui interprète le mieux et de manière simple la sécurité en termes de protection de la vie privée dans le cadre des calculs multiparties.

Des définitions formelles du modèle semi-honnête, dans le cas d'un calcul multipartie, ont été proposées par Goldreich dans [Golo4]. Dans ce qui suit, nous allons nous intéresser en particulier à l'une d'entre elles qui est une simple extension des définitions qu'on peut trouver dans le domaine des preuves sans divulgation de connaissances. Une autre définition du modèle semi-honnête est aussi donnée dans [Golo4] qui reprend la méthodologie d'émulation d'un modèle idéal.

Grosso modo, un protocole multipartie calcule une fonction à m entrées et m sorties f (par souci de simplicité, on ne s'intéresse ici qu'aux fonctionnalités déterministes) tout en protégeant les données privées des différentes parties, si tout ce que peut obtenir un ensemble de parties semi-honnêtes (p. ex., une coalition de τ parties semi-honnêtes) après leur participation au protocole, peut être obtenu à partir de leurs seules données en entrée $(x_1, x_2, \cdots, x_\tau)$ et des résultats de f disponibles au niveau de ces parties à savoir $f_i(x_1, x_2, \cdots, x_m)$ (pour i variant de 1 à τ).

Définition 5. *Soit $\mathcal{X} = \{\mathcal{X}_n\}_{n\in\mathbb{N}}$ et $\mathcal{Y} = \{\mathcal{Y}_n\}_{n\in\mathbb{N}}$ deux distributions de variables aléatoires $\mathcal{X}$ et $\mathcal{Y}$ paramétrées par des mots de longueur n. On dit que ces deux ensembles sont calculatoirement indistinguables et on note $\mathcal{X} \overset{c}{\equiv} \mathcal{Y}$, si pour toute famille de circuits booléens de taille polynômiale, $\{C_n\}_{n\in\mathbb{N}}$, pour tout polynôme positif $p(.)$ et pour tout entier n suffisamment grand,*

$$|Pr[C_n(\mathcal{X}_n) = 1] - Pr[C_n(\mathcal{Y}_n) = 1]| < \frac{1}{p(n)}. \tag{3.5}$$

Définition 6. *(Protection des données privées dans un modèle semi-honnête) :*
Soit $f : (\{0,1\}^*)^m \rightarrow (\{0,1\}^*)^m$ une fonctionnalité m-aire, où $f_i(x_1, \cdots, x_m)$ représente le $i^{ème}$ élément de $f(x_1, \cdots, x_m)$. Pour tout $I = \{i_1, \cdots, i_\tau\}$, $f_I(x_1, \cdots, x_m)$ représente la séquence $f_{i_1}(x_1, .., x_m)$,.., $f_{i_\tau}(x_1, .., x_m)$. Soit π un protocole m-parties calculant f. On note par $VUE_i^\pi(\bar{x}) = \{x_i, r_i, m_1, \cdots, m_k\}$ la vue de l'$i^{ème}$ partie durant l'exécution du protocole π sur $\bar{x} = (x_1, \cdots, x_m)$, où r_i représente l'aléa généré au niveau de la $i^{ème}$ partie et m_i le $i^{ème}$ message qu'elle reçoit. Pour tout $I = \{i_1, \cdots, i_\tau\}$, on définit la vue $VUE_I^\pi(\bar{x}) = (I, VUE_{i_1}^\pi(\bar{x}), \cdots, VUE_{i_\tau}^\pi(\bar{x}))$.

On dit que π calcule f en protégeant les données privées des différentes parties, s'il existe un algorithme probabiliste exécuté en un temps polynômial noté S, tel que pour tout I :

$$\{S(I,(x_{i_1},\cdots,x_{i_\tau}),f_I(\bar{x}))\}_{\bar{x}\in(\{0,1\}^*)^m} \stackrel{c}{\equiv} \{VUE_I^{\pi}(\bar{x})\}_{\bar{x}\in(\{0,1\}^*)^m}. \tag{3.6}$$

Remarque 1. *Il faut noter que la vue des parties dans I peut être simulée en se basant uniquement sur leurs données privées et leurs résultats de calcul. Autrement dit, la vue $VUE_I^{\pi}(\bar{x})$ comprend uniquement les vues locales des parties appartenant à la collusion I et ne comprend pas les messages échangés entre les parties honnêtes, ce qui suppose l'existence d'un canal sécurisé entre chaque paire de parties. Cependant, on peut aussi considérer le cas où la vue comprend aussi bien les vues de toutes les parties corrompues ainsi que l'ensemble des messages envoyés à travers tous les canaux de communication.*

3.2.1.2 Modèle d'attaquant malveillant

Contrairement au modèle d'attaquant semi-honnête, lorsqu'on considère un adversaire malfaisant, les utilisateurs peuvent ne pas suivre correctement les étapes du protocole. Les parties peuvent refuser de participer au protocole, quitter prématurément l'exécution du protocole, fausser les résultats intermédiaires, etc.

On peut prouver la sécurité d'un protocole multipartie dans ce modèle suivant la même démarche qu'on a suivie dans le contexte d'un modèle semi-honnête (c'est-à-dire, en utilisant le paradigme de simulation modèle idéal/réel). Pour plus de détails sur la définition du modèle d'attaquant malveillant, se référer aux travaux de Godreich dans [Golo4].

Il est démontré que tout protocole prouvé sécurisé contre des utilisateurs semi-honnêtes peut être modifié en un protocole sécurisé contre des adversaires malfaisants. Ceci est généralement réalisé en utilisant des preuves sans divulgation de connaissances qui obligent les différentes parties à suivre convenablement le protocole. Durant toutes les étapes du protocole, chaque partie utilise des preuves sans divulgation de connaissances afin de prouver aux autres qu'elle suit le protocole sans tricher.

3.3 *Primitives cryptographiques préservant la vie privée*

Nous nous intéressons dans cette section aux primitives cryptographiques utilisées dans la sécurisation des protocoles de calcul multiparties et en particulier au chiffrement homomorphe et à l'envoi superposé.

Généralement, les systèmes de chiffrement (cf. définition 7) fournissent des méthodes de transformation d'un message, appelé texte clair, en un autre message, appelé texte chiffré, en

utilisant une clé secrète. Si le système est sûr, alors le texte chiffré peut être rendu publique sans crainte, et aucune entité sauf celles possédant le secret ne peut déchiffrer le texte chiffré. Si la clé secrète utilisée pour le chiffrement est la même que celle utilisée pour le déchiffrement alors, le système de chiffrement est dit symétrique. Si les clés de chiffrement et de déchiffrement sont différentes alors, on parle d'un système de chiffrement asymétrique ou à clé publique.

Définition 7. *(définition 1.1 extraite de [Sti05])*
Un système de chiffrement est un 5-uplet $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, qui satisfait les conditions suivantes :

1. *$\mathcal{P}$ est un ensemble fini de textes clairs ;*
2. *$\mathcal{C}$ est un ensemble fini de textes chiffrés ;*
3. *$\mathcal{K}$ est un ensemble fini de clés ;*
4. *pour tout $k \in \mathcal{K}$, il existe une règle de chiffrement $e_k \in \mathcal{E}$ et la règle de déchiffrement correspondante $d_k \in \mathcal{D}$. $e_k : \mathcal{P} \to \mathcal{C}$ et $d_k : \mathcal{C} \to \mathcal{P}$ sont des fonctions telles que $d_k(e_k(m)) = m$ pour tout $m \in \mathcal{P}$.*

La sécurité des systèmes cryptographiques de manière générale et des systèmes de chiffrement en particulier est définie par rapport à de nombreuses exigences de sécurité et suivant différents modèles d'attaquant. Ces définitions sont nécessaires pour prouver la sécurité des schémas cryptographiques proposés et pour analyser les protocoles cryptographiques qui utilisent ces schémas.

En 1949, Claude Shannon publia un article intitulé "Communication Theory of Secrecy Systems" [Sha49]. Dans cet article, l'auteur introduit la notion de sécurité parfaite. Il y a confidentialité parfaite si un attaquant avec une puissance de calcul illimitée n'obtient aucune information sur le texte clair en observant le texte chiffré. Cette notion a été formalisée par Shannon en termes de distributions de probabilité de la manière suivante.

Définition 8. *Soit un système cryptographique $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$. Ce système cryptographique assure une confidentialité parfaite si*

$$Pr[x|y] = Pr[x] \text{ pour tout } x \in \mathcal{P} \text{ et } y \in \mathcal{C}.$$

C'est-à-dire que la probabilité a posteriori que le texte clair soit x, étant donné le texte chiffré y, est identique à la probabilité a priori que le texte soit x.

Cependant, les objectifs de sécurité de cette définition sont très difficiles à atteindre. Dans la pratique, on utilise une version moins forte basée sur un adversaire avec une puissance de calcul limitée polynomialement appelée *sécurité sémantique* [GM84]. Celle-ci est équivalente à d'autres définitions de sécurité telles que la sécurité polynomiale (aussi connue sous le nom de IND-CPA ou indistinguabilité des chiffrés).

La sécurité sémantique est souvent définie comme étant un objectif de sécurité à atteindre dans le cadre d'une attaque passive (c'est-à-dire que l'adversaire ne peut qu'écouter les communications). Autrement dit, l'attaquant tente d'obtenir la clé ou le texte clair à partir des chiffrés en ayant comme possibilité de chiffrer un nombre polynomial de fois des textes clairs différents. Par contre, elle n'offre aucune garantie de confidentialité si on considère un adversaire actif qui peut injecter des messages dans le réseau ou bien influencer sur le comportement des nœuds du réseau. Afin de tenir compte des attaques actives, Rackoff et Simon [RS92] ont défini la notion de sécurité contre des *attaques adaptatives par textes chiffrés choisis*. En effet, si un attaquant peut injecter des textes chiffrés dans le réseau, il peut obtenir certaines informations sur les textes clairs correspondants à travers ses interactions avec les nœuds du réseau. A cet effet, Rackoff et Simon ont modélisé ce type d'attaque en permettant à l'attaquant d'accéder au déchiffrement de textes chiffrés de son choix à l'exception bien sûr du challenge à décrypter. En d'autres termes, l'attaquant est supposé avoir accès à une entité de déchiffrement appelée *l'oracle de déchiffrement* qui peut déchiffrer un certain nombre de messages chiffrés en dehors du message à décrypter lui-même. On distingue deux variantes d'une telle attaque :

- *IND-CCA1*. L'attaquant a accès à l'oracle de déchiffrement avant qu'on lui présente le message à décrypter. L'attaque se fait donc en deux étapes : durant la première phase, l'attaquant peut accéder à l'oracle de déchiffrement et durant la seconde phase on lui retire l'oracle de déchiffrement et on lui présente le message à décrypter ;
- *IND-CCA2*. Dans ce cas, l'oracle de déchiffrement n'est pas retiré après avoir présenté à l'attaquant le message à décrypter.

Une autre notion de sécurité contre les attaques actives, appelée *non-malléabilité* a été proposée dans [DDN91]. Dans ce cas aussi l'attaquant a un accès à l'oracle de déchiffrement, mais son but n'est pas d'obtenir des informations sur le texte clair, mais de tenter d'obtenir d'autres chiffrés à partir d'un ou plusieurs textes chiffrés.

3.3.1 Chiffrements homomorphes

Le chiffrement homomorphe appartient à une classe spéciale de fonctions de chiffrement. Il permet la réalisation d'opérations algébriques sur des données chiffrées sans avoir besoin d'aucune information sur la fonction de déchiffrement.

L'utilisation des systèmes cryptographiques homomorphes dans la préservation de la vie privée a été introduite par Rivest et *al.* dans [RAD78]. Les systèmes cryptographiques homomorphes sont utilisés comme briques de base dans la construction de nombreux protocoles destinés à la préservation de la vie privée lors de calculs multiparties.

Un système de chiffrement homomorphe est un système de chiffrement entre deux structures algébriques qui transforme certaines des opérations internes à la structure algébrique des textes clairs $\mathcal{P}$, vers des opérations internes à la structure algébrique des textes chiffrés $\mathcal{C}$. Comme la plupart des systèmes cryptographiques sont définis dans des structures algébriques de type groupe ou anneau telles que $\mathbb{Z}_n$ ou $\mathbb{Z}_n^*$, nous nous intéressons donc aux définitions des chiffrements homomorphes qui transforment les opérations entre deux groupes et ceux qui transforment les opérations entre deux anneaux.

Définition 9. *Un chiffrement homomorphe est un chiffrement dans lequel l'ensemble des textes clairs $\mathcal{P}$ et l'ensemble des textes chiffrés $\mathcal{C}$ ont tous les deux une structure de groupe et pour tout $k \in \mathcal{K}$, pour toute paire de textes chiffrés $c_1 = e_k(m_1)$ et $c_2 = e_k(m_2)$, la condition suivante est vérifiée :*

$$d_k(c_1 \odot c_2) = m_1 \odot m_2.$$

$\odot$ représente la loi de composition interne définie dans $\mathcal{P}$ et $\mathcal{C}$

Définition 10. *Un chiffrement algébriquement homomorphe est un chiffrement dans lequel l'ensemble des textes clairs $\mathcal{P}$ et l'ensemble des textes chiffrés $\mathcal{C}$ ont une structure d'anneau et pour tout $k \in \mathcal{K}$, pour toute paire de chiffrés $c_1 = e_k(m_1)$ et $c_2 = e_k(m_2)$, les deux conditions suivantes sont vérifiées :*

1. $d_k(c_1 \odot c_2) = m_1 \odot m_2$,
2. $d_k(c_1 \oplus c_2) = m_1 \oplus m_2$.

$\odot$ et $\oplus$ représentent les de lois définies dans les deux anneaux $\mathcal{P}$ et $\mathcal{C}$

La sécurité des chiffrements homomorphes contre des attaques actives de type IND-CCA2 n'est pas vérifiée. Ceci peut être facilement vérifié du fait que les cryptosystèmes homomorphes sont malléables. En effet, comme l'attaquant a accès à l'oracle de déchiffrement même après avoir reçu le challenge $e_k(m)$, il peut par exemple chiffrer une constante c connue, utiliser la propriété homomorphe du chiffrement utilisé pour calculer $e_k(m) \oplus e_k(c) = e_k(m+c)$ et l'envoyer à l'oracle de déchiffrement pour obtenir $m+c$ et ainsi obtenir m. Par ailleurs, il n'a pas été donné jusqu'à présent de preuves génériques remettant en cause la sécurité des chiffrements homomorphes contre des attaques actives de type IND-CCA1 et on peut donc dire que la sécurité d'un chiffrement homomorphe dépend de la sécurité de la technique de chiffrement utilisée.

3.3.1.1 Exemples de chiffrements homomorphes

Plusieurs systèmes cryptographiques homomorphes ont été proposés dans la littérature [RSA78, EG85, Pai99, DJ01, NS98]. Dans cette section, nous présentons les trois systèmes les plus connus : les systèmes cryptographiques RSA, Paillier et ElGamal. Quant aux chiffrements complètement homomorphes (algébriquement homomorphes), le premier a été proposé en 2007 par Craig Gentry en utilisant la cryptographie basée sur les réseaux [Gen09].

Le chiffrement RSA

Le système RSA utilise l'arithmétique de $\mathbb{Z}_n$, La description du système est la suivante. Alice calcule une paire de clés publique/privée. Pour ce faire, elle choisit deux grands nombres premiers p et q tels que $n = pq$, un entier e tel que le plus grand commun diviseur de e et $\phi(n) = (p-1)(q-1)$ soit égal à un ($pgcd(e, \phi(n) = 1)$ et enfin un entier d qui est l'inverse de e modulo $\phi(n)$ ($ed \equiv 1 \ mod \ \phi(n)$). La clé publique sera constituée de n et e ; la clé privée de p, q et d. Pour envoyer la version chiffrée d'un message m à Alice, Bob calcule $c = m^e \ mod \ n$. Afin d'extraire le message en clair, Alice calcule $c^d \ mod \ n$ qui d'après le théorème d'Euler est exactement égal à m.

Propriétés homomorphes

Le système cryptographique RSA est homomorphe suivant l'opération de multiplication modulo n. Soient deux textes chiffrés $c_1 = m_1^e \ mod \ n$ et $c_2 = m_2^e \ mod \ n$, alors

$$c_1 c_2 \ mod \ n = m_1^e m_2^e \ mod \ n$$

$= (m_1 m_2)^e \ mod \ n$

est le chiffrement du produit $m_1 m_2$.

La sécurité du système RSA repose sur l'hypothèse que la fonction $e(m) = m^e \ mod \ n$ est une fonction à sens unique donc qu'il est impossible à un attaquant de décrypter un texte chiffré. La trappe que Alice cache pour pouvoir déchiffrer est la factorisation $n = pq$. Comme Alice connaît cette factorisation, elle peut calculer $\phi(n) = (p-1)(q-1)$ et calculer l'exposant de déchiffrement d en utilisant l'algorithme d'Euclide étendu.

Le problème du chiffrement RSA est qu'il est déterministe et de ce fait il ne peut pas assurer un niveau de sécurité sémantique (c'est-à-dire que la sécurité du système RSA n'est pas IND-CPA). Une version probabiliste de RSA a été proposée, mais elle n'est malheureusement pas homomorphe.

Le chiffrement d'ElGamal

Le chiffrement ElGamal repose sur le problème du logarithme discret. La description du système est la suivante. Avant toute chose, Alice va générer une paire de clés publique/privée. Pour cela, elle choisit un grand nombre premier p et un générateur g du groupe cyclique $\mathbb{Z}_p^*$. Afin de constituer sa paire de clés, Alice sélectionne un nombre aléatoire a dans $\mathbb{Z}_{p-1}$ et calcule $h = g^a$ dans $\mathbb{Z}_p^*$ et considère $(g, p-1, h)$ comme étant sa clé publique et a comme étant sa clé privée. Pour envoyer la version chiffrée d'un message m à Alice, Bob choisit aléatoirement $k \in \mathbb{Z}_{p-1}$ et calcule le chiffré $(c_1, c_2) = (g^k, mh^k)$ dans $\mathbb{Z}_p^*$. Enfin, Alice obtient m en calculant $c_2 (c_1^a)^{-1}$.

Propriétés homomorphes

Dans le système ElGamal, la multiplication de deux textes chiffrés est homomorphe, il permet aussi la multiplication d'un texte chiffré par une constante et l'élévation d'un texte chiffré à la puissance d'une constante. Soient deux textes chiffrés (c_1, c_2) et (d_1, d_2) de deux messages m_1 et m_2, en utilisant respectivement les deux valeurs aléatoires y_1 et y_2, alors

$(c_1 d_1, c_2 d_2) = (g^{y_1} g^{y_2}, (m_1 h^{y_1})(m_2 h^{y_2}))$

$= (g^{y_1 + y_2}, m_1 m_2 h^{y_1 + y_2})$

est le chiffrement de $m_1 m_2$. De plus, soit une constante k, alors

$$(c_1, kc_2) = (g^{y_1}, km_1 h^{y_1})$$

est le chiffrement de km_1, et enfin

$$(c_1^k, c_2^k) = (g^{y_1 k}, m_1^k h^{y_1 k})$$

est le chiffrement de m_1^k.

Contrairement à RSA, le chiffrement ElGamal est IND-CPA. En effet, Il est prouvé que si le problème DDH est difficile dans le groupe G choisi (c'est-à-dire, si (g^x, g^y, g^{xy}) et (g^x, g^y, g^r) pour $r \in G$, sont indistinguables) alors, le chiffrement ElGamal est sémantiquement sûr.

Le chiffrement de Paillier

Un autre chiffrement homomorphe très intéressant et dont la sécurité est prouvée IND-CPA a été proposé par Paillier dans [Pai99]. Il est décrit comme suit : Alice calcule une paire de clés publique/privée : elle choisit tout d'abord un entier $n = pq$ où p et q sont de grands nombres premiers tels que $pgcd(n, \phi(n)) = 1$ et considère un groupe $G = \mathbb{Z}_{n^2}^*$ d'ordre k. Elle suppose aussi l'existence de $g \in G$ d'ordre n. Sa clé publique sera constituée de n et g et sa clé privée sera quant à elle composée de p et q. Afin de chiffrer un message $m \in \mathbb{Z}_n$, Bob choisit aléatoirement un entier $r \in \mathbb{Z}_n^*$ et calcule $c = g^m r^n \, mod \, n^2$. Pour extraire le message en clair, Alice calcule le logarithme discret de $c^{\lambda(n)} \, mod \, n^2$ pour obtenir $m\,\lambda(n) \in \mathbb{Z}_n$, où $\lambda(n)$ représente la fonction de Carmichael. Comme $pgcd(\lambda(n), n) = 1$, Alice peut aisément calculer $\lambda(n)^{-1} \, mod \, n$ et obtient ainsi m.

Propriétés homomorphes

Le système cryptographique de Paillier permet l'addition et la soustraction homomorphe de deux textes chiffrés, l'addition et la soustraction entre un texte chiffré et une constante, et la multiplication d'un chiffré par une constante. Soit $c_1 = g^{m_1} r_1^n \; mod \; n^2$ et $c_2 = g^{m_2} r_2^n \; mod \; n^2$. Alors,

$$c_1 c_2 \; mod \; n^2 = g^{m_1} r_1^n g^{m_2} r_2^n = g^{m_1 + m_2} r_1 r_2^n \; mod \; n^2$$

est le chiffrement de $m_1 + m_2$,

$$c_1 g^k \ mod \ n^2 = g^{m_1} r_1^n g^k = g^{m_1+k} r_1^n \ mod \ n^2$$

est le chiffrement de $m_1 + k$, et

$$c_1^k \ mod \ n^2 = (g^{m_1} r_1^n)^k = g^{km_1} (r_1^k)^n \ mod \ n^2$$

est le chiffrement de km_1. Les opérations de soustraction entre des textes chiffrés et entre un texte chiffré et une constante peuvent être obtenues en calculant respectivement $c_1 c_2^{-1} \ mod \ n^2$ et $c_1 g^{-k} \ mod \ n^2$.

3.3.2 Envoi superposé (DC-Networks)

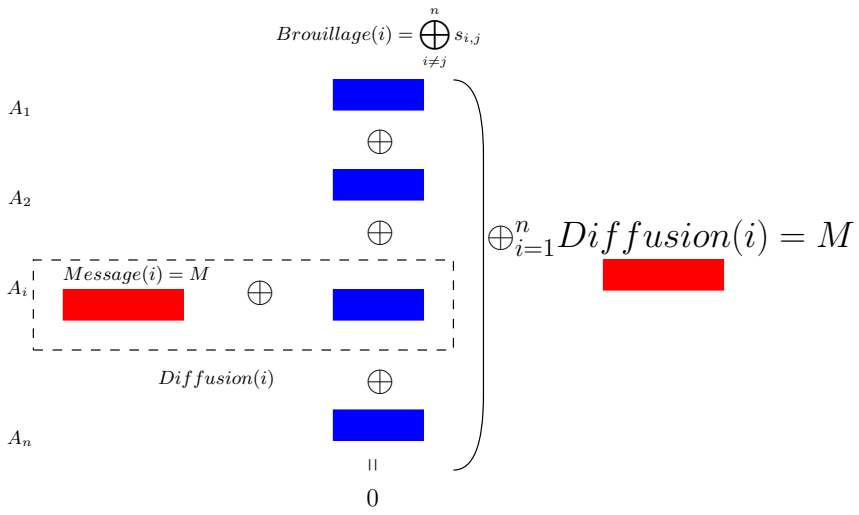

FIGURE 3.7 – Principe de l'envoi superposé

Le problème du dîner des cryptographes et sa solution ont été proposés par David Chaum en 1988 [Cha88]. Pour décrire ce protocole, il a utilisé une parabole dans laquelle trois cryptographes se trouvent à une même table et veulent savoir si quelqu'un a payé, sans pour autant savoir lequel d'entre eux a réellement payé (c'est-à-dire que quelqu'un dise "J'ai payé" avec anonymat d'émission).

Il est utilisé dans les réseaux informatiques pour permettre aux utilisateurs d'envoyer ou de recevoir des messages de manière anonyme quelle que soit l'architecture du réseau utilisé. Les réseaux implémentant ce protocole sont appelés les réseaux du dîner des cryptographes ou DC-nets, et le protocole est connu sous le nom de protocole DC-net (Dining Cryptographers Network) ou protocole d'envoi superposé.

L'envoi superposé est mis en place dans un réseau synchronisé, dans lequel, à chaque top d'horloge, chaque participant diffuse un message. L'idée de base est que les utilisateurs se mettent d'accord pour tout émettre, même quand ils n'ont rien à transmettre, en utilisant un format tel que la combinaison de toutes les émissions produit :

- un résultat nul, si aucun utilisateur n'a essayé de transmettre des informations utiles,
- le message d'un utilisateur si un seul d'entre eux a essayé de transmettre,
- des messages brouillés si plusieurs utilisateurs ont essayé de transmettre en même temps.

Le déroulement du protocole est le suivant :

Envoi Superposé

Soit un ensemble d'utilisateurs $A_1, ..., A_n$. Supposons que chaque couple d'utilisateurs (A_i, A_j) possède un secret commun, $s_{ij} = s_{ji}$ de l-bits.
Chaque utilisateur A_i encode un message $Message(i)$ en une chaîne de l-bits à zéro s'il n'a rien à transmettre, ou bien en un message de l-bit s'il a quelque chose à transmettre.

Déroulement du tour :

1. Chaque utilisateur A_i diffuse :
   $Diffusion(i) = Brouillage(i) \oplus Message(i)$ où $Brouillage(i) = \bigoplus_{j=1, j\neq i}^{n} s_{ij}$.
2. Chaque utilisateur calcule le résultat du tour :
   $S = \bigoplus_{i=1}^{n} Diffusion(i)$.

Comme chaque s_{ij} apparaît sous deux formes (une fois inséré par A_i (s_{ij}) et une deuxième fois inséré par A_j (s_{ji})), les brouillages s'annulent mutuellement et chaque utilisateur obtient S= $\bigoplus_{i=1}^{n} Message(i)$.

Exemple 5. *Un exemple d'envoi superposé dans un réseau constitué de trois nœuds est décrit dans la figure 3.8. Avant de diffuser son message (*110101*) aux autres nœuds du réseau, le nœud A_2 calcule le XOR de ce message avec les clés secrètes préalablement échangées avec les nœuds A_1 et A_3 (resp.*

101011 *et* 110110*). Les nœuds* A_1 *et* A_3 *font de même avec leurs messages* (000000*). Tous les nœuds calculent le XOR de tous les messages reçus et obtiennent tous le même message à savoir celui de* A_1*. Ceci s'explique par le fait que les clés secrètes apparaissent deux fois dans le calcul du XOR global.*

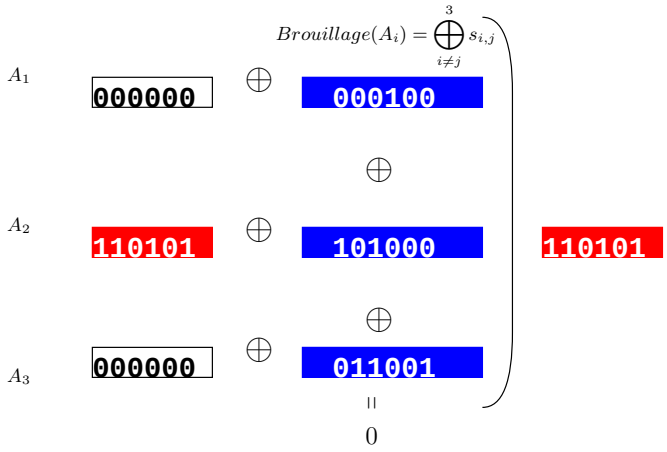

FIGURE 3.8 – Exemple d'envoi superposé avec trois participants. $s_{1,2} = 101011$, $s_{2,3} = 110110$ et $s_{1,3} = 101111$.

La version du protocole DC-net proposée initialement par David Chaum et décrite ci-dessus est définie dans le corps fini F_2. Cependant, ce protocole peut être défini dans n'importe quel groupe abélien fini (cf. [Wai90]) :

Soit $P = \{P_1, P_2, \cdots, P_n\}$ l'ensemble des participants, et soit $(F, +)$ un groupe abélien fini. Le *graphe de clés* G est défini comme étant un graphe non orienté et sans boucle où P représente l'ensemble de ses sommets. À chaque tour d'exécution du protocole, les participants P_i, P_j qui sont voisins dans G (c'est-à-dire les participants directement liés dans le graphe G) choisissent aléatoirement dans F un secret commun $s_{ij} = s_{ji}$. Chaque participant P_i choisit un message $Message(i) \in F$ et diffuse la somme locale :

$$Diffusion(i) = Message(i) \oplus \sum_{(P_i,P_j)\in G} signe(i-j) \cdot s_{ij} \tag{3.7}$$

et reçoit la somme globale :

$$S = \sum_{i=1}^{n} Diffusion(i). \tag{3.8}$$

Puisque chaque secret s_{ij} apparaît sous deux formes (une fois inséré par P_i (s_{ij}) et une deuxième fois inséré par P_j (s_{ji})), on obtient :

$$S = \sum_{i=1}^{n} Message(i). \tag{3.9}$$

Sécurité de l'envoi superposé

Si les secrets s_{ij} sont choisis aléatoirement et renouvelés à chaque tour, l'envoi superposé offre un anonymat d'émission parfait (inconditionnel). Un attaquant n'obtient aucune information sur l'émetteur d'un message même s'il contrôle tous les canaux de communication. Il est prouvé dans [Cha88] que la seule chose que l'attaquant peut obtenir à propos des messages transmis par les utilisateurs est la parité de leurs diffusions, car le contenu des messages est protégé par un masque jetable. L'anonymat de réception inconditionnelle peut aussi être assurée si l'on considère un mécanisme de diffusion fiable.

Dans [Cha88], la sécurité inconditionnelle de l'envoi superposé a été prouvée dans le corps fini F_2. Dans [Wai90], le lemme suivant est prouvé dans un groupe abélien fini F quelconque :

Lemme 1. *(lemme 2.2 [Wai90])*

Soit I l'ensemble des participants contrôlés par un attaquant, et supposons que le graphe $G \setminus (P \times I)$ (autrement dit, le sous-graphe induit par l'ensemble $P \setminus I$) est connexe. Soit $(Diffusion(1), \cdots, Diffusion(n)) \in F^n$ le résultat d'un tour. Alors pour chaque vecteur $(Message(1), \cdots, Message(n)) \in F^n$ qui, conformément aux connaissances a priori de l'attaquant à propos des $Message(i)$ satisfait

$$\sum_{i=1}^{n} Diffusion(i) = \sum_{i=1}^{n} Message(i), \tag{3.10}$$

le même nombre de combinaisons de clés (qui satisfont l'équation 3.7 conformément aux connaissances a priori de l'attaquant à propos des s_{ij}) existe.

Par conséquent, la probabilité conditionnelle de $(Message(1), \ldots, Message(n))$ étant donné $(Diffusion(1), \ldots, Diffusion(n))$ est égale à la probabilité conditionnelle de $(Message(1), \ldots, Message(n))$ étant donné uniquement la somme globale 3.10.

On peut constater d'après ce lemme que la sécurité de l'envoi superposé dépend du graphe de clé initial G. En effet, le graphe de clés doit être construit de telle sorte qu'il reste connexe en présence d'une collusion. Par exemple, si le graphe de clés est complet, le graphe

sera résistant à une collusion composé de $n-1$ participants. Dans le cas d'un graphe non complet, la résistance du protocole va dépendre du schéma de connexité du graphe de clés initial.

Considérations pratiques

La sécurité parfaite de l'envoi superposé repose sur le fait que les s_{ij} ne soient utilisés qu'une seule fois et qu'ils ne soient pas prévisibles.

La proposition de David Chaum était que les utilisateurs échangent des disques optiques avec de grandes quantités de bits aléatoires pour former les s_{ij} . Avec cette solution, on peut émettre anonymement autant de bits qu'il y a dans le disque optique. Dans la pratique, la solution la plus généralement adoptée est celle où les utilisateurs n'échangent pas des s_{ij} mais des graines G_{ij} qui, avec des Générateurs de Nombres Pseudo-Aléatoires (GNPAs) [GGM86, MvOV01, BK06], permettent de générer un s_{ij} à chaque tour.

Si on utilise des générateurs de nombres pseudo-aléatoires, la sécurité du protocole va reposer plutôt sur la sécurité du générateur et ne sera plus inconditionnelle comme dans le protocole de base. Cependant, la simplification apportée par l'utilisation de ces générateurs et le fait que de nos jours la sécurité soit souvent basée sur le même type d'hypothèses, font que cette dégradation de la sécurité est généralement acceptée.

3.4 *Cadre de travail*

Notations

Nous donnons ici les notations utilisées dans notre protocole.

s_{ij}	Clé secrète de m *bits* partagée entre deux agents B_i et B_j.
$\|s_{ij}\|$	Le nombre de bits de s_{ij}
$Rand_{seed}$	GNPA basé sur la graine *seed*
pk	Une clé publique
sk	Une clé privée
Enc_{pk}	La fonction de chiffrement homomorphe utilisant la clé publique pk
Dec_{sk}	La fonction de déchiffrement homomorphe utilisant la clé privée sk
$Enc_{pk}(m;r)$	Le chiffré du message m obtenu en utilisant un chiffrement homomorphe probabiliste avec un aléa r

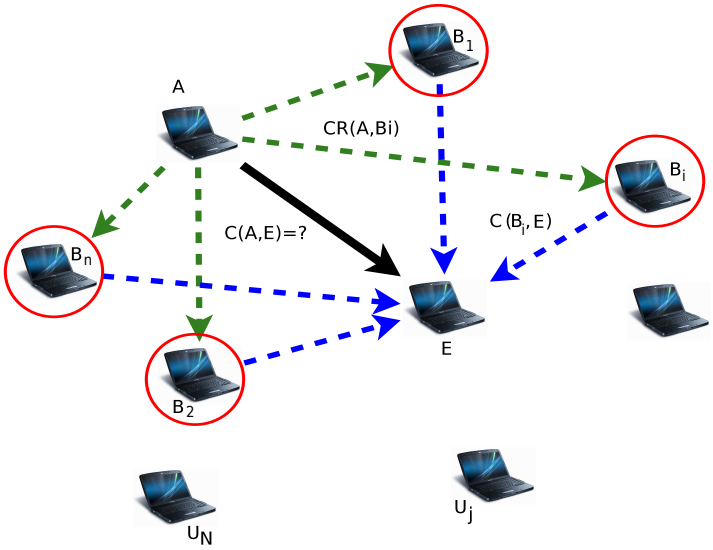

FIGURE 3.9 – Définition du problème

Objectifs

On suppose que chaque utilisateur du système est représenté par un *agent* (de manière plus formelle par une machine de Turing), qui a une capacité de calcul suffisante et qui a la capacité de communiquer avec d'autres agents.

Notre problème est le suivant (figure 3.9) : un agent demandeur A veut interagir avec un nouveau partenaire (un autre agent) E, mais A n'a pas suffisamment d'informations sur ce partenaire. En effet, si A n'a jamais interagi avec E par le passé (absence d'un historique des interactions ou d'expériences passées entre A et E), il n'aura de ce fait aucune ou peu de connaissances préalables au sujet du comportement de E qui lui permettrait de déduire une mesure de confiance significative sur la fiabilité de E.

Afin de remédier à cela, il fait appel aux méthodes d'établissement des valeurs de confiance et de réputation. A entame, suivant une métrique appropriée à l'application, un processus d'établissement de confiance afin de calculer la valeur de confiance ou de réputation qui lui permettrait de décider d'interagir ou non avec E.

Dans un système de confiance ou de réputation décentralisé, A consulte un groupe d'agents témoins, qu'on note ici $\{B_1, \cdots, B_n\}$, de manière directe ou indirecte (par transitivité), qui sont censés avoir une valeur de confiance fonctionnelle concernant E.

Si on reprend les notations utilisées dans la section 3.1.4, le problème est défini de la manière suivante :

Définition 11. *Soit un agent A qui veut calculer le degré de confiance qu'il veut accorder à un autre agent E et soit un ensemble $B = \{B_1, \cdots, B_n\}$ de témoins qui connaissent E. La formule donnant la valeur de confiance de E, selon le point de vue de A, est :*

$$C(A,E) = 1/n \sum_{i=1}^{n} CR(A, B_{i,1}) \left(\prod_{j=1}^{k} CR(B_{i,j}, B_{i,j+1}) \right) C(B_{i,k+1}, E) \tag{3.11}$$

où, $C(A,B)$ représente la valeur de confiance fonctionnelle qu'un agent A accorde à un autre agent B ou bien une valeur par défaut ; $CR(A,B)$ représente la valeur de confiance en les recommandations de B selon le point de vue de A ou bien une valeur par défaut.

Dans le cas où la longueur des chemins de confiance est égale à un, on peut utiliser la version réduite de l'équation précédente à savoir :

$$C(A,E) = 1/n \sum_{i=1}^{n} CR(A, B_i) C(B_i, E). \tag{3.12}$$

Dans le cas où la transitivité n'est pas prise en compte, on se retrouve avec un système de réputation basique dont la formule donnant la valeur de réputation est :

$$C(A,E) = 1/n \sum_{i=1}^{n} C(B_i, E). \tag{3.13}$$

On suppose que les différentes valeurs de confiance considérées ici sont des scalaires. De plus, par souci de simplicité et de compréhension, on va s'intéresser tout au long de ce chapitre à l'équation 3.12. Les protocoles qu'on va proposer ou étudier dans ce chapitre peuvent être généralisés ou réduits aux deux autres équations.

Le calcul de $C(A,E)$ peut être vu comme un protocole multipartie qui calcule la fonction m-aire et déterministe $f((x_1, \cdots, x_n)(y_1, \cdots, y_n)) = \sum_{i=1}^{n} x_i y_i$ où $x_i = CR(A, B_i)$ et $y_i = C(B_i, E)$ représentent les données privées en entrée de f.

Modèle d'attaquant

Afin d'analyser la sécurité de notre proposition, on suppose l'existence d'un adversaire externe s'exécutant en un temps polynômial probabiliste, contrôlant un sous-ensemble fixe d'agents.

Notre premier objectif est de proposer une solution permettant de calculer la fonction f tout en protégeant les données privatives des agents dans le cadre d'un modèle semi-honnête. De manière plus formelle :

Définition 12. *Soit $f(\bar{x}) = \sum_{i=1}^{n} x_i y_i$ avec $\bar{x} = ((x_1, \cdots, x_n)(y_1, \cdots, y_n))$ retournant un seul résultat. Soit π le protocole $(n+1)$-parties calculant f. On dit que π calcule f tout en protégeant $\bar{x}$, si pour tout attaquant contrôlant une collusion $I = \{A, B_{i_1}, \cdots, B_{i_\tau}\}$, les seules données qu'il peut obtenir après l'exécution de π sont celles qu'il obtient à partir de $f(\bar{x})$, $(x_1, \ldots, x_n)$ et $(y_1, \cdots, y_\tau)$, et pour tout attaquant contrôlant une collusion $I = \{B_{i_1}, \cdots, B_{i_\tau}\}$, les seules données qu'il peut obtenir après l'exécution de π sont celles qu'il obtient à partir de $f(\bar{x})$ et $(y_1, \cdots, y_\tau)$.*

Notre deuxième objectif est de permettre que cette solution soit facilement extensible pour qu'elle puisse résister à des comportements malveillants. Plusieurs comportements malveillants qui ne sont pas englobés dans un modèle semi-honnête peuvent survenir dans un système de gestion de confiance. Par exemple :

- insertion de valeurs de confiance erronées. En effet, si on suppose que les valeurs de confiance sont comprises entre deux valeurs (si $C(B_i, B_j) \in [L_1, L_2]$), un agent malveillant peut insérer des données en dehors de cet intervalle ;
- l'agent demandeur A peut s'aider de valeurs spéciales pour connaître les données privatives des témoins.

Une collusion constituée de $n-1$ témoins (c'est-à-dire $\{A, B_{i_1}, \cdots, B_{i_{n-1}}\}$) malveillants ou semi-honnêtes contre un seul agent honnête isolé, est difficile ou quasiment impossible à éviter par un protocole multipartie sécurisé. En effet, un agent demandeur malveillant A peut choisir, de manière déterministe, $n-1$ agents corrompus et un agent non corrompu. On peut contourner ce problème en imposant aux agents une méthode de sélection de témoins spécifique. L'idée est de choisir une méthode de sélection probabiliste qui assure la présence d'un grand nombre de témoins non corrompus.

Lemme 2. *([PRT04])*
Soit $N > 1$ le nombre de témoins potentiels et soit $0 < n < N$ le nombre de témoins participant au processus de calcul de confiance. Soit $\tau < N$ le nombre d'agents corrompus dans N. Si les agents sont uniformément distribués à travers N, alors il existe un schéma de sélection de témoins qui garantit qu'au moins deux témoins sont non corrompus avec une probabilité supérieure à $(1 - \frac{1}{n})(\frac{N-\tau-1}{N-1})$.

Démonstration. Considérons le schéma de sélection de témoins suivant. A choisit un premier témoin B_1. Chaque témoin choisi, sélectionne un autre témoin avec une probabilité $1 - 1/n$. Soit B_h le premier témoin honnête choisi et soit m le nombre de témoins corrompus choisis avant B_h. La probabilité que B_h choisisse un témoin non corrompu est donc $P \geq (1 - 1/n)(\frac{N-\tau-1}{N-m-1}) \geq (1 - 1/n)(\frac{N-\tau-1}{N-1})$. □

On trouve dans la littérature d'autres schémas de sélection similaires qui s'appuient sur des techniques telles que l'élection de leaders et le tirage aléatoire collectif. Pour plus de détails, se référer aux travaux suivants [Sak89, Fei99].

3.5 *Étude des travaux existants*

Il existe deux méthodes de calcul et de dérivation des valeurs de confiance pouvant être utilisées dans notre cas. La première est basée sur le calcul de sommes multiparties qui sont souvent utilisées dans les systèmes de réputation, la deuxième sur le calcul de produits scalaires multiparties donnant lieu à des systèmes de confiance plus complexes.

Dans cette section, nous allons étudier quelques protocoles multiparties existants permettant soit le calcul d'une addition multipartie, soit le calcul d'un produit scalaire multipartie tout en préservant les données privatives de ces parties. Autrement dit, sécuriser le calcul des fonctions 3.12 et 3.13.

3.5.1 Système de confiance basé sur une somme multipartie sécurisée

Il existe plusieurs protocoles proposés dans la littérature qui permettent le calcul d'une addition multipartie avec préservation de la vie privée (Private Multi-Party Summation Protocol (PMPS)). Un premier exemple simple est celui proposé dans [CKV+02].

Somme multipartie : Protocole 1.

---

1. Étape d'initialisation : l'agent demandeur A, ordonne de manière circulaire les agents témoins ($A \rightarrow B_1 \rightarrow B_2 \rightarrow \ldots \rightarrow A$), envoie à chaque témoin d'indice i l'identité de son successeur dans le cercle à savoir $i+1$ et envoie au témoin B_n son identité.
2. A choisit aléatoirement un entier $r \neq 0$ et l'envoie au témoin B_1.
3. En recevant s_p de leurs prédécesseurs dans le cercle, chaque agent calcule $s_p + y_i$ avec $y_i = C(B_i, E)$ (la note de réputation (donnée privée) du témoin B_i) et l'envoie à son successeur dans le cercle.
4. Quand A reçoit à son tour la somme calculée par son prédécesseur B_n, il soustrait r et obtient la somme $\sum_{i=1}^{n} y_i$. Il peut ainsi calculer la note de réputation globale suivant la formule utilisée.

Ce protocole permet effectivement de ne rien révéler sur les données privées des différents témoins tant qu'il n'y a pas de collusion. Cependant, il suffit que deux témoins B_{i-1} et B_{i+1} décident de former une collusion contre B_i pour révéler la valeur exacte de y_i,

et ce, en soustrayant la somme envoyée par B_{i-1} à celle reçue par B_{i+1}. Par conséquent, ce protocole ne respecte pas complètement le modèle semi-honnête.

En ce qui concerne la complexité de ce protocole, $O(n)$ messages sont échangés entre les agents et les calculs réalisés ont un coût négligeable.

Un deuxième protocole qui améliore un peu cette proposition et qui est utilisé comme brique de base dans le schéma de Yao et *al.* (cf. section 3.5.2), a été proposé dans [AEDS03]. En utilisant les mêmes notations que dans le premier protocole, nous pouvons décrire son fonctionnement comme suit :

Somme multipartie : Protocole 2.

---

1. Chaque témoin B_i choisit un nombre aléatoire r_i.
2. Chaque agent B_{2i} envoie à l'agent B_{2i+1} la somme $y_{2i} + r_{2i}$.
3. Chaque B_{2i+1} envoie à l'agent B_{2i} le nombre aléatoire r_{2i+1}.
4. Les parties impaires calculent $\sum_{i=0}^{n-1} y_i + r_i$ et les parties paires calculent $\sum_{i=0}^{n-1} r_i$ en utilisant, par exemple, une approche basée sur un arbre de calcul.
5. A la fin de l'exécution du protocole, les parties impaires (resp., paires) échangent simultanément leurs résultats et obtiennent $\sum_{i=0}^{n-1} y_i$ (les auteurs n'ont pas précisé exactement le protocole d'échange de secrets utilisé).

Ce protocole nécessite $O(n)$ communications. Cependant, d'après la proposition 1, il ne protège pas complètement les données privées des agents lorsque ces derniers se comportent de manière semi-honnête.

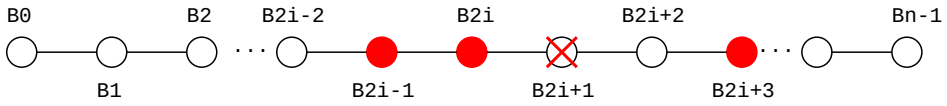

FIGURE 3.10 – Attaques contre les agents impairs

Proposition 1. *Il suffit qu'un attaquant contrôle une collusion composée de l'agent demandeur A et de trois témoins pour obtenir la valeur de la donnée privée y_i associée à un agent témoin B_i.*

Démonstration. Pour prouver cette proposition, nous allons donner deux instances d'attaque. La première contre l'ensemble des agents témoins pairs et la deuxième attaque contre

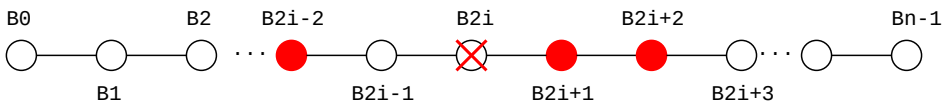

FIGURE 3.11 – Attaques contre les agents pairs

l'ensemble des agents témoins impairs. Nous présentons un exemple de chaque attaque.

Attaques sur le protocole 2

1. **Si un attaquant veut connaître la valeur de la donnée privée** y_{2i+1}, il crée une collusion $I = \{A,\ B_{2i+3}, B_{2i},\ B_{2i-1}\}$:
   - B_{2i+3} fournit : $X = \sum_{i=1}^{2i+1}(y_i + r_i)$.
   - $Y = \sum_{i=1}^{2i-1}(y_i + r_i)$ est fourni par B_{2i-1}.
   - $Z = (y_{2i} + r_{2i})$ et r_{2i+1} sont fournis par B_{2i}.
   - Et enfin on obtient : $y_{2i+1} = X - (Y + Z + r_{2i+1})$.
2. **Si maintenant un attaquant veut connaître la valeur de la donnée privée** y_{2i}, il crée une collusion $I = \{A, B_{2i+1},\ B_{2i+2},\ B_{2i-2}\}$. Au niveau de B_{2i+1} on a $y_{2i} + r_{2i}$: il coopère avec B_{2i-2} et B_{2i+2} dans le but d'obtenir r_{2i} et par conséquent il obtient y_{2i} :
   - B_{2i+2} fournit : $X = r_1 + ... + r_{2i-2} + r_{2i-1} + r_{2i} + r_{2i+1}$.
   - $Y = r_1 + ... + r_{2i-2} + r_{2i-1}$ est fourni par B_{2i-2}.
   - $Z = r_{2i+1}$ est fourni par B_{2i+1}.
   - on obtient : $r_{2i} = X - (Y + Z)$ et par conséquent, on retrouve y_{2i}.

□

Essayons de voir maintenant de plus près une troisième proposition [PRT04] qui améliore la sécurité des deux premières solutions proposées ci-dessus.

Somme multipartie : Protocole 3.

1. Étape d'initialisation : A envoie aux agents témoins $\{B_1, \cdots, B_n\}$ le détail sur les autres agents participant au calcul (l'identité des agents témoins ainsi que son identité) et choisit un nombre aléatoire r_A.

2. Chacun des $n+1$ agents participant au protocole divise sa donnée privée y_i en $n+1$ fragments suivant la procédure suivante : l'agent i choisit n nombres aléatoires $r_i^1, \cdots, r_i^n$ et calcule $r_i = y_i - \sum_{j=1}^{n} r_i^j$. Il sauvegarde r_i et envoie $r_i^1, \cdots, r_i^n$ aux n autres agents, tels que chaque agent k reçoit le fragment r_i^k.
3. Chaque agent k calcule $s_k = \sum_{i=1}^{n} r_i^k + r_k$ et envoie s_k à l'agent demandeur A.
4. A calcule, une fois qu'il a reçu tous les s_i (pour $i = 1$ à n) de la part des n agents témoins, $\sum_{k=1}^{n+1} s_k - r_A$.

On peut facilement remarquer que ce protocole préserve parfaitement les données privées des différents agents quand ces derniers se comportent de façon semi-honnête et qu'il est particulièrement optimal en termes de résistance aux collusions. En d'autres termes, ce protocole assure la protection des données privées quel que soit l'attaquant contrôlant une collusion composée de τ témoins semi-honnêtes avec $\tau \in [1, n-1]$.

Concernant les performances de ce protocole, il nécessite une complexité de communication plus élevée que celle des deux premiers protocoles. Elle est estimée à $O(n^2)$ messages.

L'inconvénient des propositions qu'on a décrites jusqu'à maintenant est que, suivant le protocole en question, elles permettent bien la préservation des données privées dans un modèle semi-honnête, mais elles sont difficilement extensibles pour pouvoir être aussi résistantes aux attaques lorsque les agents se comportent de manière malveillante.

A cet effet, plusieurs solutions utilisant des primitives cryptographiques telles que le chiffrement homomorphe (très utile dans la vérification et la validité des données échangées durant l'exécution d'un protocole cryptographique) ont été proposées pour d'une part avoir des solutions dont la sécurité peut être prouvée dans un modèle semi-honnête et d'autre part pouvoir étendre les solutions proposées afin qu'elles soient aussi sécurisées dans un modèle d'attaquant malveillant.

Dans [PRT04], les auteurs proposent un protocole qui s'appuie sur deux primitives cryptographiques homomorphes à savoir le schéma de mise en gage basé sur le logarithme discret (discrete-log commitment scheme) et le schéma de partage de secret de Shamir [Sha79]. Leur proposition respecte le modèle semi-honnête, est optimale en termes de résistance aux collusions et permet de prendre en considération quelques comportements malfaisants tels que la validité des valeurs de confiance échangées. De manière plus précise, elle vérifie si les valeurs appartiennent bien à l'intervalle de notation. Par exemple, si on suppose que les notes de réputation doivent être comprises dans l'intervalle $[1, 100]$ et que A sélectionne 4 témoins. Dans le cas où un témoin adopte un comportement malveillant et

fournit une note de 400, A va se retrouver avec une somme supérieure à 400 par exemple et donc inutilisable.

Somme multipartie : Protocole 4.

On suppose que les valeurs de réputation y_i fournies par les agents témoins B_i sont des entiers appartenant au groupe G_q où q est l'ordre premier du groupe. On suppose aussi l'existence d'un canal sécurisé entre chaque paire d'agents.

1. Étape d'initialisation : A choisit un groupe G_q, deux générateurs de ce groupe g et h avec $log_g h$ difficile à calculer. Il envoie à chaque agent témoin $\in \{B_1, \cdots, B_n\}$: g, h ainsi que le détail sur lui-même et sur les autres agents témoins participant au processus.
2. Chaque témoin d'indice i choisit deux polynômes de degré n : $p^i(x) = p_0^i + p_1^i x + \cdots + p_n^i x^n$ et $q^i(x) = q_0^i + q_1^i x + \cdots + q_n^i x^n$ tels que $p_0^i = y_i$ et que les autres coefficients soient choisis aléatoirement dans G_q.
3. B_i envoie à chaque agent j, $j = 1, \ldots, i-1, i+1, \ldots, n+1$ de l'ensemble $\{A, B_1, \ldots, B_{i-1}, B_{i+1}, \ldots, B_n\}$, le point $p^i(j)$ et $q^i(j)$ et diffuse la mise en gage des coefficients de ses deux polynômes : $g^{p_0^i} h^{q_0^i}, \ldots, g^{p_n^i} h^{q_n^i}$.
4. Une fois qu'un témoin m reçoit : $p^1(m), \ldots, p^{m-1}(m), p^{m+1}(m), \ldots, p^n(m)$ et $q^1(m), \ldots, q^{m-1}(m), q^{m+1}(m), \ldots, q^n(m)$, calcule $p^m(m)$, $q^m(m)$, $s_m = \sum_{i=1}^n p^i(m)$, $t_m = \sum_{i=1}^n q^i(m)$ et envoie s_m et t_m à A.
5. A calcule $s_{n+1} = \sum_{i=1}^n p^i(n+1)$ et $t_{n+1} = \sum_{i=1}^n q^i(n+1)$.
6. Une fois que A aura reçu les $s_1, \cdots, s_n$ et $t_1, \cdots, t_n$, il construit le polynôme $s(x) = \sum_{i=1}^n p^i(x)$ et obtient $s(0)$ qui correspond à la somme des notes de réputation. Pour ce faire, il calcule $\sum_{i=1}^{n+1} s_i L_i(0)$, où $L_i(0)$ est le polynôme de Lagrange au point 0 qui peut être calculé ainsi : $L_i(0) = \prod_{j=1, j\neq i}^{n+1} \frac{j}{j-i}$.

Durant les étapes 5 et 6, les agents peuvent vérifier que les messages qu'ils reçoivent des autres agents sont valides, en exploitant la propriété homomorphe des mises en gage ($g^{p_0^i} h^{q_0^i}, \ldots, g^{p_n^i} h^{q_n^i}$) qu'ils s'échangent durant l'étape 3. Pour chaque paire ($p^i(m)$, $q^i(m)$) reçue, l'agent m calcule $g^{p^i(m)} h^{q^i(m)} = \prod_{j=0}^n (g^{p_j^i} h^{q_j^i})^{m^j}$ pour vérifier qu'ils correspondent bien aux vrais coefficients de $p^i(x)$ et $q^i(x)$, qu'il a préalablement diffusés à travers les mises en gage. De plus, grâce aux mises en gage qu'il reçoit et en particulier $g^{p_0^i} h^{q_0^i}$, l'agent demandeur A peut vérifier, en utilisant par exemple des preuves sans divulgation de connaissances, la validité des valeurs de réputation reçues. Par exemple, vérifier si $p_0^i \in [-L, +L]$ où $-L$ et $+L$ sont respectivement la valeur de notation minimale et maximale autorisée.

L'inconvénient de ce protocole est qu'il nécessite trop d'échanges de messages entre les différents agents. Sa complexité en communication est estimée à $O(n^3)$ messages.

3.5.2 Système de confiance basé sur un produit scalaire multipartie sécurisé

Le travail le plus récent sur la protection des données privées dans les systèmes de confiance a été proposé par Yao et *al.* dans [YTP07]. Les auteurs de cet article proposent de sécuriser le calcul du produit scalaire multipartie défini dans l'équation 3.12. Leur protocole peut être généralisé au calcul de la fonction définie dans l'équation 3.11 pour prendre en considération des systèmes de gestion de confiance plus complexes. Ils proposent, à cet effet, un protocole cryptographique utilisant un chiffrement homomorphe et la somme multipartie décrite dans le protocole 2 présenté ci-dessus.

Leur solution est présentée comme étant une extension aux travaux effectués auparavant dans le cas de la sécurisation des produits scalaires à deux parties [AD01, GLLM04]. Le déroulement de ce prococole peut être résumé suivant ces étapes :

Protocole de Yao et *al.*

- En utilisant un chiffrement homomorphe (section 3.3.1) et probabiliste, A chiffre une à une ses valeurs privées x_i ($Enc_{p_k}(x_i; r_i)$) et les envoie aux B_i ;
- Chaque B_i calcule l'exponentielle y_i du chiffrement reçu de la part de A comme suit (voir propriétés du chiffrement homomorphe) : ($Enc_{pk}(x_i y_i - v_i; r_i^{y_i} r_i^{'})$), masque le résultat, et envoie le résultat masqué à A ;
- Les B_i suivent le protocole 2 afin de calculer la somme $\sum_{i=1}^{n} V_i$ et envoient leurs résultats à A ;
- A calcule la somme des résultats masqués reçus de la part des B_i et lui soustrait le résultat du protocole 2.

Le fait que ce protocole se déroule en deux étapes :

- masquer les valeurs privatives des agents témoins,
- calculer la somme de ces masques de manière sécurisée,

le rend non optimal en termes de coûts de communication et de calcul. De plus, comme nous l'avons démontré précédemment, le protocole d'addition multipartie (protocole 2)

utilisé introduit quelques vulnérabilités dues aux collusions de certains participants. Afin de remédier à ce problème, on pourrait proposer l'utilisation d'un protocole d'addition multipartie plus résistant aux collusions tel que le schéma de sommation décrit dans le Protocole 3. Cependant, cette approche génère d'une part un flux de communication supplémentaire de $O(n^2)$ messages, d'autre part, elle est difficilement extensible au modèle d'attaquant malveillant.

3.6 *Notre proposition*

Le protocole de Yao et *al.* s'exécute en deux phases. La première phase est basée sur un chiffrement homomorphe et engendre un produit scalaire masqué par un nombre aléatoire. Durant la deuxième phase, les agents utilisent le protocole 2 pour calculer ce masque qui est ensuite soustrait au résultat de la première phase. À la fin de ce processus, toutes les parties (ou seulement l'initiatrice du calcul (A)) obtiennent le résultat du produit scalaire non masqué.

Au-delà des problèmes de performance qu'engendre l'utilisation du protocole 2, le protocole de Yao et *al.* souffre d'un problème de sécurité beaucoup plus contraignant. En effet, ce protocole est vulnérable à une collusion composée de l'agent demandeur A et trois témoins honnêtes mais curieux.

Afin de remédier à ces problèmes, nous proposons un nouveau schéma inspiré de l'envoi superposé. Celui-ci permet l'envoi d'un message à partir d'un ensemble d'utilisateurs synchronisés et coopératifs sans connaître l'utilisateur émetteur du message, et ce, même en présence d'une collusion entre les membres de cet ensemble. En nous basant sur cette capacité de DC-net à résister aux collusions, nous obtenons un schéma de calcul de Produits Scalaires Multipartie Résistant aux Collusions (PSM-RC) constituées de $\tau \in [1, n-1]$ agents témoins. De plus, notre solution est facilement évolutive et plus efficace en termes de coûts de communication, en réduisant de 50% les coûts par rapport à la version proposée par Yao et *al.*.

Par ailleurs, nous utilisons le même chiffrement homomorphe que celui utilisé dans le protocole de Yao et *al.* : chaque agent calcule une partie du produit scalaire en utilisant les propriétés du chiffrement homomorphe et en masquant le résultat. Une différence importante est que dans le cas de Yao et *al.*, le masque est choisi aléatoirement et que dans notre cas il est calculé en utilisant le même principe que le protocole DC-net. Ceci permet aux masques de s'annuler dès la première étape et d'obtenir immédiatement le résultat final.

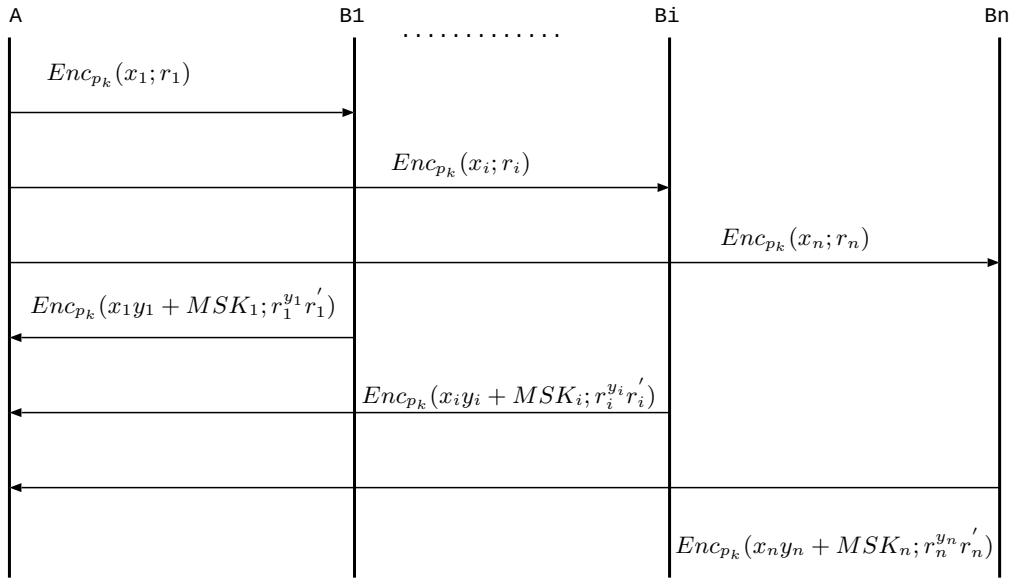

FIGURE 3.12 – Messages échangés dans PSM-RC

3.6.1 Description

Nous présentons dans cette section la version générique de notre protocole.

On suppose que $X = (x_1, ..., x_n) \in \mathbb{Z}_m^n$, $Y = (y_1, ..., y_n) \in \mathbb{Z}_m^n$ et que toutes les opérations sont réalisées modulo m. Le chiffrement homomorphe utilisé est randomisé et possède les deux propriétés suivantes (on peut utiliser par exemple le chiffrement de Paillier (cf. section 3.3.1.1)) :

- $Enc_{p_k}(x; r).Enc_{p_k}(y; r') = Enc_{p_k}(x + y; r.r')$,
- $Enc_{p_k}(x; r)^y = Enc_{p_k}(x.y; r^y)$.

L'agent demandeur A ordonne les agents témoins et envoie à chacun d'entre eux l'identité de tous les autres agents participant au protocole. Soit $B = \{B_1, B_2, \cdots, B_n\}$ l'ensemble des agents témoins et soit le *graphe de clés* G défini comme étant un graphe non orienté et sans boucle où B représente l'ensemble de ses sommets. À chaque round d'exécution du protocole, chaque couple d'agents (B_i, B_j) voisins dans G, choisit aléatoirement dans $\mathbb{Z}_m$ une clé secrète commune ($s_{ij} = s_{ji}$). Le reste des opérations à réaliser par A et les agents témoins est résumé ci-dessous.

Version générique du PSM-RC

- Pour i **variant de** 1 **à** n

1. L'agent demandeur A génère un couple de clés privée/publique (s_k, p_k) et envoie p_k aux témoins B_i (ceci peut être effectué juste une fois ou de façon périodique).
2. A chiffre de manière homomorphe les éléments x_i de son vecteur X avec sa clé publique. Le texte chiffré obtenu $c_i = Enc_{p_k}(x_i; r_i)$ est envoyé aux B_i où r_i est une séquence de bits aléatoires.
3. Comme on utilise un chiffrement homomorphe, même si B_i ne connaît pas la clé privée du chiffrement, il est capable de calculer le chiffrement correspondant à $x_i y_i$, malgré qu'il ne connaisse pas x_i. B_i calcule le chiffrement w_i ainsi :
   - $w_i = c_i^{y_i} \bmod m = Enc_{p_k}(x_i.y_i; r_i^{y_i})$.
4. Afin de masquer y_i, B_i calcule à la manière du protocole DC-net :
   - Le masque $MSK_i = \sum_{(B_i,B_j)\in G} sign(i-j) \cdot s_{ij}$,
   - Le chiffrement $w_i^{'} = w_i \ . \ Enc_{p_k}(MSK_i; r_i^{'}) = Enc_{p_k}(x_i.y_i + MSK_i; r_i^{y_i} r_i^{'})$.
5. B_i envoie $w_i^{'}$ à l'agent demandeur A.

- Quand A reçoit tous les chiffrements $w_i^{'}$, il calcule :
$\prod_{i=1}^{n} w_i^{'} \bmod m = \prod_{i=1}^{n} Enc_{p_k}(x_i.y_i + MSK_i; r_i^{y_i}.r_i^{'}) = Enc_{p_k}(\sum_{i=1}^{n} x_i y_i + \sum_{i=1}^{n} MSK_i; \prod_{i=1}^{n} r_i^{y_i}.r_i^{'})$
$= Enc_{p_k}(X.Y + \sum_{i=1}^{n} MSK_i; \prod_{i=1}^{n} r_i^{y_i}.r_i^{'})$. $(*)$

- A utilise sa clé privée s_k ainsi que la fonction de déchiffrement Dec afin de déchiffrer $(*)$. Comme chaque clé s_{ij} apparaît sous deux formes (une fois insérée par B_i (s_{ij}) et une seconde fois insérée par B_j (s_{ji})), elle s'annulent deux à deux et la somme $\sum_{i=1}^{n} MSK_i$ vaudra zéro, et par conséquent A obtient immédiatement le résultat final $X.Y$.

Afin de mettre en place les secrets s_{ij}, nous pouvons faire appel aux systèmes cryptographiques à clés publiques, qui sont des mécanismes faciles à mettre en œuvre. Le mécanisme à clés publiques utilisé dans notre protocole pour l'échange des secrets s_{ij} est le protocole de Diffie Hellman (DH) [DH76]. Chaque B_i initie, à chaque tour, un nouvel échange de DH avec chacun de ses voisins agents B_j afin d'établir une nouvelle clé secrète s_{ij}.

Comme dans le reste de la littérature sur le sujet, nous nous plaçons dans le cadre d'un modèle semi-honnête. Ainsi, nous ne prenons pas en compte les attaques actives et plus particulièrement celles du type *homme du milieu*. Le passage du modèle semi-honnête au modèle d'attaquant malveillant est évoqué dans la section 3.6.2. Pour se

protéger contre de telles attaques, il est possible d'utiliser des approches spécifiques comme l'utilisation de mécanismes d'établissement de certificats adaptés aux réseaux ad hoc [ZH99, LKZ+04, KZL+01, HBC01, CBH03] ou l'utilisation de la cryptographie basée sur les identités (cf. [LP06]).

Une amélioration très intéressante qu'on peut préconiser afin d'éviter les coûts relativement excessifs engendrés par des échanges de clés de DH est de lancer cet échange une seule fois, à l'initialisation du protocole. Ainsi, chaque paire d'utilisateurs (B_i, B_j) utilise sa clé secrète, générée à l'initialisation du protocole, comme graine ($seed_{ij}$) pour les futurs générations de clés secrètes. En d'autres termes, à chaque tour, chaque paire (B_i, B_j) génère de manière pseudo aléatoire une clé commune s_{ij}. En effet, en choisissant un groupe adéquat, les clés secrètes générées durant les étapes de négociation de clés de DH auront toutes les conditions nécessaires pour constituer une graine à un générateur de nombres pseudo aléatoires. Cependant, afin d'assurer les propriétés de sécurité du type "*forward secrecy*", on pourrait envisager une méthode de gestion du *seed* et des secrets générés comme celle utilisée dans le protocole *TLS* (Transport Layer Security) qui est connu pour avoir cette propriété.

3.6.2 Analyse de sécurité

Notre protocole est basé sur deux primitives principales à savoir un chiffrement homomorphe sémantiquement sûr et le protocole DC-net. Intuitivement et de manière informelle, on peut affirmer que la sécurité de notre protocole dépend essentiellement de la sécurité de ces deux primitives. En effet, on peut facilement remarquer que le chiffrement homomorphe est le garant de la confidentialité des données privées (x_i) de l'agent demandeur A et l'envoi superposé est quant à lui utilisé afin de masquer les données privées des différents témoins B_i (les valeurs y_i).

Par ailleurs, comme le graphe de clés est construit en utilisant le protocole de Diffie-Hellman, la sécurité de notre proposition va aussi dépendre de la sécurité de ce protocole. Cependant, le graphe de clés peut être construit suivant d'autres approches connues pour être très sûres. Par exemple, dans la version originelle du protocole d'envoi superposé [Cha88], Chaum proposait l'utilisation de disques optiques avec de grandes quantités de bits aléatoires pour former les secrets s_{ij}.

Cas 1 : Modèle d'attaquant semi-honnête

Nous allons tout d'abord essayer d'analyser la sécurité de notre protocole, qu'on note ici π, par rapport au modèle d'attaquant semi-honnête défini dans la définition 12. On distingue deux cas de figure selon que l'agent A est corrompu par l'attaquant ou non.

Les preuves de sécurité de notre proposition sont données dans les théorèmes 1 et 2

Théorème 1. *Soit un attaquant représenté par une machine de Turing probabiliste à temps polynomial contrôlant un ensemble fixe d'agents* $C = \{A\} \cup I$ *(*$I = \{B_{i_1}, \cdots, B_{i_\tau}\}$ *où* $\tau \in [1, n-1]$*). Si le sous-graphe de clés* $G \setminus (B \times I)$ *engendré est connexe alors, le protocole* π *est sécurisé dans le cadre du modèle semi-honnête (cf. définition 6).*

Démonstration. Puisque $A \in C$, l'attaquant peut accéder à toutes les données chiffrées par le chiffrement homomorphe choisi. Par ailleurs, comme le sous graphe $G \setminus (B \times I)$ est connexe alors, d'après le lemme 1, la probabilité conditionnelle de $(x_1y_1, x_2y_2, \cdots, x_ny_n)$ sachant $(x_iy_i + \sum_{(B_i,B_j)\in G} signe(i-j) \cdot s_{ij})_{\forall i \in [1..n]}$ est égale à la probabilité conditionnelle de $(x_1y_1, x_2y_2, \cdots, x_ny_n)$ étant donné uniquement la somme $\sum_{i=1}^{n} x_iy_i$.

Par conséquent, l'attaquant n'obtient aucune information supplémentaire à propos de $(y_1, y_2, \cdots, y_n)$ en dehors de celles qu'obtient l'attaquant à partir de $(I, (x_1, \ldots, x_n), (y_{i_1}, \ldots, y_{i_\tau}), \sum_{i=1}^{n} x_iy_i)$.

Nous pouvons donc affirmer que pour tout algorithme probabiliste exécuté en un temps polynômial noté S :

$$\{S(I, (x_1, \cdots, x_n), (y_{i_1}, \cdots, y_{i_\tau}), f_I(\bar{x}))\} \stackrel{c}{\equiv} \{VUE_I^{\pi}((x_1, x_2, \cdots, x_n), (y_1, y_2, \cdots, y_n))\}$$

et que de ce fait le protocole respecte bien le modèle semi-honnête. □

Théorème 2. *Soit un attaquant représenté par une machine de Turing probabiliste à temps polynomial contrôlant un ensemble d'agents* $C = \{B_{i_1}, \cdots, B_{i_\tau}\}$ *où* $\tau \in [1, n-1]$*) alors,* π *est sécurisé dans le cadre du modèle semi-honnête (cf. définition 6).*

Démonstration. Contrairement au cas où A faisait partie de la collusion, la vue de l'attaquant est réduite ici aux informations privées des agents membres de la collusion et aux messages chiffrés envoyés par les autres agents. On peut donc affirmer que la sécurité du protocole va dépendre de la sécurité du chiffrement homomorphe choisi. Si on suppose que le chiffrement homomorphe choisi est, dans le pire des cas, sémantiquement sûr, l'attaquant ne peut extraire aucune information en temps polynômial à propos d'un x_i ou d'un y_i à partir des différents chiffrés en dehors de celle qu'il aurait obtenue sans ces chiffrés. Par conséquent on peut affirmer, d'après la définition du modèle semi-honnête (définition 12), que le protocole respecte bien le modèle semi-honnête, et ce, quels que soient les agents ou le nombre d'agents qui composent la collusion. □

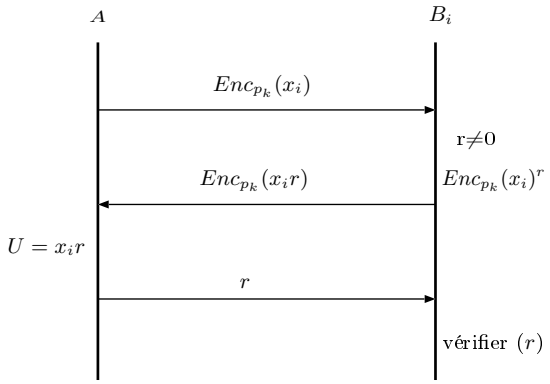

FIGURE 3.13 – Exemple d'un schéma de vérification des données du protocole

Cas 2 : Modèle d'attaquant malveillant

Jusqu'à maintenant nous avons considéré la sécurité de notre protocole dans un modèle d'attaquant semi-honnête et nous avons prouvé qu'effectivement notre schéma résiste bien à ce modèle. Cependant, notre schéma tel qu'il est présenté ci-dessus ne permet pas de faire face à certains comportements malfaisants de certains agents.

Les agents peuvent en effet adopter un comportement malfaisant en fournissant par exemple des données erronées, ce qui a pour conséquence de biaiser le résultat final et ainsi gaspiller inutilement la bande passante et le temps processeur de tous les participants au protocole. Par exemple, un témoin pourrait insérer des notes de confiance non autorisées ou encore, l'agent demandeur pourrait utiliser des valeurs spéciales telles que $X = (x_1 = 0, \cdots, x_{i-1} = 0, x_i = 1, x_{i+1} = 0, \cdots, x_n = 0)$ pour obtenir la valeur de y_i, etc.

Il a été prouvé par Goldreich dans [Golo4] que tout protocole cryptographique prouvé sûr dans le cadre d'un modèle d'attaquant semi-honnête peut, en utilisant notamment les preuves sans divulgation de connaissance, être étendu à une version qui soit aussi sûre lorsque les participants au protocole se comportent de manière malveillante.

Grâce notamment à l'utilisation, dans notre protocole, d'un schéma homomorphe, on peut aisément intégrer certaines preuves sans divulgation de connaissance basées sur ces schémas qui permettent de résister à certains comportements malveillants. Par exemple, tester la validité de certaines valeurs telles que les valeurs de confiance échangées entre les agents. À cet effet, nous nous sommes intéressés à l'attaque décrite ci-dessus (c'est-à-dire, au cas où l'agent demandeur tenterait d'envoyer $X = (0, \cdots, 0, 1, 0, \cdots, 0)$). Le détail d'une preuve très simple est présenté dans la figure 3.13). Il est à notre avis très important de

considérer cette attaque, car elle permet à l'agent A d'avoir un accès direct aux données privées des autres agents et remet même en question la sécurité du protocole dans le modèle semi-honnête.

Bien sûr, la même démarche peut être suivie pour traiter les autres comportements malveillants. Cependant, se protéger contre un comportement générique en utilisant les techniques proposées dans [Golo4] est complexe et coûteux. Une description complète de cette transformation va au delà des objectifs de ce manuscrit.

3.6.3 Produit Scalaire Multipartie K-Résistant aux Collusions (PSM-K-RC)

Nous avons pu constater lors de l'analyse de sécurité de la version générique de notre protocole que plus le degré minimal du graphe de clés augmente, plus la sécurité du protocole est meilleure.

Une première idée est donc de choisir un graphe de clés complet (c.-à-d. un graphe de clés possédant $n(n-1)/2$ arrêtes). On obtiendrait dans ce cas une sécurité optimale (une résistance optimale aux collusions), mais en revanche on obtiendrait, lors de la mise en place des clés secrètes, une très grande complexité en termes du nombre de communications et en termes de calcul. Ce coût serait approximativement égal à $O(n^2)$. Ceci peut entraîner des problèmes d'évolutivité si on veut l'appliquer à de gros réseaux.

Afin d'optimiser les performances de notre protocole et ainsi diminuer les coûts en termes de communication et de calcul, nous allons utiliser un graphe de clés moins dense. En d'autres termes, chaque agent ne va partager des clés secrètes qu'avec un certain nombre d'agents amis. Il existe plusieurs manières de construire ce graphe.

Nous avons opté ici pour une approche complètement distribuée et spontanée qui correspond mieux aux réseaux autonomes tels que les réseaux ad hoc et qui dépend d'un paramètre de sécurité k qu'on peut ajuster suivant la capacité du réseau d'accueil.

Afin de construire ce graphe de manière déterministe, chaque agent B_i choisit aléatoirement k agents distincts B_j (pour j variant de 1 à k). De ce fait, chaque sommet du graphe de clés aura une densité moyenne d approximativement égale à $2k$, car la probabilité qu'un agent B_i soit choisi par plus de k autres agents est très faible (cf. proposition 2).

Proposition 2. *La probabilité qu'un agent B_i soit choisi par plus de δk autres agents, pour $\delta > 1$, est de l'ordre de $O(e^{-\delta k})$.*

Démonstration. Un agent choisit k autres agents parmi les $n-1$ participant au protocole (il ne se choisit pas lui même). Sachant que les agents sont choisis aléatoirement, un agent

donné B_i a une probabilité $k/(n-1)$ d'être choisi par un autre agent B_j. Considérons maintenant la variable aléatoire X_i qui est égale à 1 si un agent est choisi et 0 dans le cas contraire. Ce qui signifie que la variable X_i suit une loi de Poisson avec une distribution $Prob[X_i = 1] = k/n - 1$. Comme chaque agent fonctionne indépendamment des autres, la suite $X_1, ..., X_{n-1}$ est une suite de variables aléatoires indépendantes avec $E[\sum_{i=1}^{n-1} X_i] = k$. En utilisant l'inégalité de Chernoff, il est prouvé que pour $\delta > 1$: $prob[\sum_{i=1}^{n-1} X_i > \delta k] < (e^{\delta-1}/\delta^{\delta})^k$. Donc, on peut déduire que la probabilité qu'un agent soit choisi par plus de δk agents est de l'ordre de $O(e^{-\delta k})$. □

Le résumé complet de cette deuxième version de notre proposition est donné ci-dessous.

L'agent demandeur A ordonne les agents témoins et envoie à chacun d'entre eux l'identité de tous les autres agents participant au protocole. Soit $B = \{B_1, B_2, \cdots, B_n\}$ l'ensemble des agents témoins et soit le *graphe de clés* G défini comme étant un graphe non orienté et sans boucle où B représente l'ensemble de ses sommets. Le reste des opérations à réaliser par A et les agents témoins est résumé ci-dessous.

PSM-K-RC

Soit g l'élément primitif qui génère le groupe $G_r = \mathbb{Z}_p^*$ d'ordre $p >> m$

A L'initialisation du protocole (round r=1) :

Pour i **de** 1 **à** n **faire** (Construction du graphe de clés G)

- B_i sélectionne aléatoirement un secret : $e_i \in G_r$.
- B_i choisi aléatoirement parmi les $n-1$ agents, un ensemble de k agents amis (voisins) différents $\{f_1, f_2, ..., f_k\}$.
- **Pour** j **de** 1 **à** k **faire**
  - B_i et f_j se mettent d'accord sur (G_r, g).
  - B_i envoie à f_j la partie publique g^{e_i}.
  - Quand f_j reçoit g^{e_i}, il :
    - calcule $(g^{e_i})^{e_{f_j}}$.
    - répond à B_i en lui envoyant à son tour sa partie publique $g^{e_{f_j}}$.
  - B_i calcule $(g^{e_{f_j}})^{e_i}$.
  - Enfin, B_i et f_j hachent $g^{e_i e_{f_j}}$ en une chaîne de m-bit : $seed_{B_i,f_j} = h(g^{e_i e_{f_j}})$ qui sera la graine secrète entre B_i et f_j.

A chaque round $r \geq 1$ **:**

Considérons $s^{r=0}_{B_i,B_j} = seed_{B_i,B_j}$.

Pour i **de** 1 **à** n **faire**

- Soit l'ensemble $BF_{B_i} = \{B_j / s^r_{B_iB_j} existe\}$.
- A génère une paire de clés privée et publique (s_k, p_k) et envoie p_k à B_i.
- A chiffre l'élément x_i de son vecteur X avec sa clé publique de façon homomorphe. Le résultat de ce chiffrement $c_i = Enc_{p_k}(x_i; r_i)$ est envoyé à B_i où r_i est une chaîne de bits aléatoires.
- Comme le chiffrement est homomorphe, B_i n'est pas obligé de connaître la clé privée de A pour chiffrer x_iy_i. B_i calcule w_i comme suit :
  - $w_i = c_i^{y_i} mod\, m = Enc_{p_k}(x_i.y_i; r_i^{y_i})$.
- B_i calcule le masque : $MSK_{B_i} = \sum_{B_j \in BF_{B_i}} signe(i-j) \cdot s^r_{B_iB_j}$ où, $s^r_{B_i,B_j} = Rand_{seed_{B_i,B_j}}(s^{r-1}_{B_i,B_j})$.
- Dans le but de cacher y_i à A, B_i calcule :
  - Le chiffrement $w_i^{'} = w_i \ .\ Enc_{p_k}(MSK_{B_i}; r_i^{'}) = Enc_{p_k}(x_i.y_i + MSK_{B_i}; r_i^{y_i}.r_i^{'})$.
- Enfin B_i envoie $w_i^{'}$ à A.

Quand A reçoit tous les $w_i^{'}$, il calcule :
$\prod_{i=1}^n w_i^{'} mod\, m = \prod_{i=1}^n Enc_{p_k}(x_i.y_i + MSK_{B_i}; r_i^{y_i}.r_i^{'}) = Enc_{p_k}(\sum_{i=1}^n x_iy_i + \sum_{i=1}^n MSK_{B_i}; \prod_{i=1}^n r_i^{y_i}.r_i^{'})$
$= Enc_{p_k}(X.Y + \sum_{i=1}^n MSK_{B_i}; \prod_{i=1}^n r_i^{y_i}.r_i^{'})$. $(*)$

A utilise la fonction Dec et sa clé privée s_k pour déchiffrer $(*)$. Comme chaque s_{B_i,B_j} apparaît sous deux formes (une fois inséré par B_i (s_{B_i,B_j}) et une fois inséré par B_j (s_{B_j,B_i})), la somme $\sum_{i=1}^n MSK_{B_i}$ sera égale à zéro, et par conséquent A obtient immédiatement le résultat final $X.Y$.

3.6.3.1 Analyse de sécurité

On peut distinguer deux stratégies pour isoler un nœud du graphe de clés. La première est de corrompre un nombre fixe de nœuds du graphe avant l'exécution du protocole. Une deuxième approche plus efficace mais plus difficile à mettre en œuvre consiste à cibler uniquement les voisins du nœud à isoler.

Dans le premier cas, si on suppose l'existence d'un attaquant représenté par une machine de Turing probabiliste à temps polynomial contrôlant un ensemble fixe de nœuds (agents) $I = \{B_{i_1}, \cdots, B_{i_\tau}\}$, la probabilité p qu'un nœud soit isolé des autres nœuds du graphe est :

$$p = \frac{\binom{\tau}{2k}}{\binom{n}{2k}} = \frac{(n-2k)!\tau!}{(\tau-2k)!n!}. \tag{3.14}$$

Le graphe présenté dans la figure 3.14 montre les différentes probabilités de réussite de cette attaque suivant la valeur de k dans un graphe constitué de 100 nœuds et dont la moitié d'entre eux sont corrompus. Ces résultats montrent clairement que la sécurité du protocole est proportionnelle à k et qu'il suffit d'une petite valeur de k pour avoir une probabilité p avoisinant zéro.

Une comparaison avec le protocole de Yao et *al.* qui peut être représenté par un graphe de degré $d = 3$, montre que pour un paramètre de sécurité très faible $k = 2$, la probabilité de réussite de l'attaque dans notre protocole est de $p \approx 0,0587$, alors que dans le protocole de Yao et *al.* cette probabilité est de $p \approx 0,1212$.

Dans le deuxième cas, il est très facile de vérifier que la sécurité du protocole dépend directement de k.

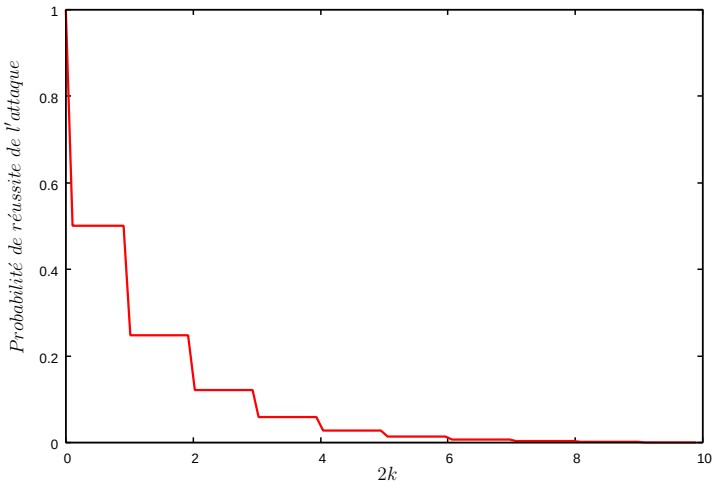

FIGURE 3.14 – Résistance aux collusions suivant k avec $n = 100$, $|I| = 50$

3.6.3.2 Analyse de performance

Malgré qu'une analyse fine du problème ne peut être faite qu'à travers l'implémentation de notre proposition dans une vraie application, nous allons essayer de donner ici quelques

comparaisons en termes de performance entre notre schéma et celui de Yao et *al.*.

Le premier tour d'initialisation de notre protocole nécessite un coût supplémentaire constant de $O(k)$, dû aux échanges de clés de Diffie-Hellman. Par exemple, pour $k = 2$, on aura seulement deux échanges de clés de Diffie-Hellman par nœud, pour une sécurité meilleure que celle du protocole de Yao et *al.*

Pour $r > 1$, le protocole de Yao et *al.* génère un total de $4n$ communications :

- L'agent demandeur A et tous les agents témoins B_i envoient $2n$ données chiffrées par tour ;
- Durant le protocole de sommation, les B_i réalisent au moins $2n$ communications par tour.

Comme notre protocole s'exécute en une seule étape, les seules données envoyées ou reçues correspondent aux $2n$ messages chiffrés échangés entre A et les agents témoins. Ceci représente une réduction de 50% des communications comparé au protocole de Yao et *al.*

Les coûts des opérations de calcul sont approximativement équivalents dans les deux modèles. Dans les deux protocoles (tableau 3.1) : A utilise n opérations de chiffrement homomorphe par tour et chaque B_i effectue une opération exponentielle par tour. Dans notre cas, le seul surcoût additionnel est celui causé par les opérations du générateur de nombres pseudo aléatoires effectués au niveau de chaque agent B_i. À chaque tour, chaque B_i génère de manière pseudo aléatoire d_i secrets. En d'autres termes, chaque tour nécessite approximativement nd_i générations pseudo aléatoires. Dans le cas du modèle de Yao et *al.*, le protocole 2 engendre n générations aléatoires par tour.

3.7 *Conclusion*

Dans ce chapitre, nous avons traité le problème de protection de la vie privée dans les systèmes de gestion de confiance et de réputation distribués. À cet effet, nous avons opté pour la protection des données privées des différents participants au système plutôt que de nous appuyer sur l'anonymisation des identités, et ce, pour les raisons que nous avons évoquées plus haut. Cette démarche s'inscrit dans le contexte de la sécurisation des protocoles de calcul multiparties.

Protocole	Opération	Complexité
Protocole de Yao et al.	Cal. (Alice) Cal. (B_i)	n op. homomorphes $log\, y_i$ op. homomorphes + op. GNPA (Protocole 2)
Notre Protocole	Cal. (Alice) Cal. (B_i)	n op. homomorphes $log\, y_i$ op. homomorphes + $O(k)$ GNPA

TABLE 3.1 – Coûts des opérations de calcul.

On distingue dans le contexte des protocoles multiparties deux modèles d'attaquant qui sont : le modèle semi-honnête et le modèle d'attaquant malveillant. Tous les protocoles proposés ou étudiés dans ce chapitre ont été évalués par rapport à ces deux modèles.

Nous avons présenté certains travaux proposés dans le cadre de la protection des données privées, d'une part, dans les systèmes de réputation, d'autre part, dans des systèmes de gestion de confiance plus génériques. On peut classer ces travaux dans deux catégories : une catégorie de travaux efficaces et résistants à un modèle d'attaquant semi-honnête, mais difficilement extensible à un modèle d'attaquant malveillant et une deuxième catégorie qui est quant à elle pensée de telle manière à ce qu'elle soit facilement extensible à un modèle d'attaquant malveillant.

Nous avons isolé dans cette seconde catégorie deux travaux intéressants.

Le premier est proposé dans [PRT04] pour protéger les notes de réputation dans un système de réputation additif basé sur le calcul d'une moyenne multipartie. Ce schéma résiste à un modèle d'attaquant semi-honnête et il est particulièrement optimal en termes de résistance aux collusions, mais malheureusement il génère trop de messages.

Le second est proposé dans [YTP07] pour protéger les recommandations dans un système de gestion de confiance basé sur le calcul d'un produit scalaire multipartie. Ce schéma, contrairement au premier, s'avère moins coûteux en termes de communication, mais n'offre qu'une sécurité partielle dans un modèle d'attaquant semi-honnête.

Afin de trouver une solution qui soit à la fois efficace et optimale en termes de résistance aux collusions, nous avons d'abord pensé à modifier le modèle proposé dans [PRT04] pour qu'il puisse être utilisé dans un système de confiance plus général. Cependant, le résultat obtenu induit toujours une communication très élevée. Ensuite, en nous inspirant du modèle

proposé par Yao et *al.*, nous avons proposé un nouveau schéma permettant le calcul d'un produit scalaire distribué qui respecte la vie privée des intervenants au calcul. Le schéma que nous avons proposé est beaucoup plus sûr dans un modèle semi-honnête comparé à celui proposé par Yao et *al.*. En effet, notre protocole permet une haute résistance aux collusions dans le sens où il suffit qu'un seul participant soit honnête pour que le protocole soit sûr dans le cadre du modèle semi-honnête.

Il est aussi plus efficace en termes de communication que ses prédécesseurs. Concernant les performances de notre protocole, il réduit de 50% les communications par rapport au protocole de Yao et *al.*.

Concernant la sécurité de notre protocole dans un modèle d'attaquant malveillant, il est établi qu'il peut aisément être généralisé pour qu'il puisse prendre en compte de nombreux comportements malveillants.

Chapitre 4

Détection de nœuds répliqués dans les réseaux de capteurs

Nous avons mis en évidence dans l'état de l'art une vulnérabilité nuisible contre les systèmes de gestion de confiance distribués : l'attaque Sybille. Cette attaque est un problème crucial dans de nombreux systèmes et semble résister jusqu'à présent à la plupart des solutions proposées. De nombreuses applications distribuées ainsi que la majorité des mécanismes de sécurité supposent que chaque entité possède exactement une seule identité et que cette dernière est unique. Lorsque cette hypothèse n'est plus vérifiée, le mécanisme de sécurité peut faire l'objet d'attaques et les résultats de l'application peuvent être corrompus.

Les mécanismes de confiance et de réputation sont des exemples types de systèmes dont le fonctionnement repose sur l'hypothèse qu'il existe une seule et unique identité pour chaque entité. Si une entité avait la possibilité de manipuler comme elle le souhaite son identité ou celle des autres, elle remettrait en cause la crédibilité des résultats obtenus par le système de confiance : des entités honnêtes pourraient être accusées à tort, des entités malhonnêtes pourraient se voir accorder des notes de confiance élevées, etc. De plus, même si une entité malveillante est détectée et pénalisée comme c'est la procédure dans de telles situations, elle n'aura qu'à changer d'identité.

Plusieurs approches ont été proposées afin de prévenir ou d'atténuer les risques d'une attaque Sybille. On trouve des approches basées sur une autorité de certification, sur la vérification de ressources, sur la vérification de position, sur des dispositifs de confiance matériels, etc. L'approche la plus efficace est celle basée sur la certification de chaque identité utilisée dans le système (cf. [Dou02]).

Lorsque le réseau utilisé n'offre pas de protection physique suffisante, comme c'est le cas dans les réseaux de capteurs, un attaquant peut aller au-delà du mécanisme d'authentification utilisé en répliquant le nœud et par conséquent, il peut répliquer le logiciel qui le compose. Cette attaque est connue dans la littérature sous l'appellation d'attaque par réplication de nœuds et elle fait l'objet de ce chapitre.

De nombreuses mesures de sécurité ont été proposées afin de remédier à ce problème. Le principe de base de toutes ces méthodes consiste à imposer à chaque nœud la déclaration de ses coordonnées géographiques soit à une autorité centrale, soit à un ensemble de nœuds du réseau appelés témoins. Chaque témoin a pour rôle de vérifier la validité des déclarations reçues (positions géographiques cohérentes, signature, etc.), sauvegarder les déclarations validées et vérifier si chaque identité existe dans une et une seule position. Dans le cas où cette dernière vérification se révèle négative, une demande de révocation des répliques concernées est initiée par le témoin.

Ces méthodes diffèrent selon l'approche choisie pour sélectionner les témoins et selon qu'elles soient centralisées ou décentralisées. En effet, sélectionner un très grand nombre de témoins influe négativement sur le rendement des capteurs, car ceci va nécessiter un très grand nombre de communications, de calculs et de taille mémoire. D'un autre côté, cela augmente la probabilité de détection des répliques. Par conséquent, l'enjeu est de trouver la meilleure approche de sélection possible qui permette, d'une part, une grande probabilité de détection et, d'autre part, un faible coût de communication, de calcul et de mémoire. De plus, la méthode de sélection de témoins doit être résistante aux attaques. En effet, si l'attaquant arrive à prédire les futurs témoins associés à une identité donnée, il pourrait les neutraliser afin que ses répliques restent indétectables. Par conséquent, un protocole de détection est dit idéalement résistant aux corruptions de témoins, s'il ne divulgue aucune information sur les futurs témoins. Autrement dit, la probabilité de choisir un nœud x comme témoin, suivant le point d'observation d'un attaquant, doit se rapprocher le plus possible de $1/n$, n étant le nombre de nœuds dans le réseau.

Dans une approche centralisée, chaque nœud du réseau envoie la liste des déclarations de position de ses voisins directs à la station de base qui a pour tâche d'examiner toutes les listes reçues afin de détecter d'éventuels conflits, ce qui occasionne une congestion très forte des routes vers la station de base. De plus, la sécurité de cette approche repose sur un seul point de vulnérabilité.

Les premiers à avoir proposé des protocoles de détection distribués de répliques sont Parno et *al.* dans [PPG05]. Leur protocole *Line Selected Multicast* (*LSM*) est celui qui donne les meilleurs résultats. Pour un réseau de n capteurs, ce protocole engendre $O(\sqrt{n})$ messages transmis ou reçus par nœud, alors que dans les approches traditionnelles les communications engendrées sont de $O(n)$ messages par nœud. Par ailleurs, chaque nœud du réseau a besoin de sauvegarder $O(\sqrt{n})$ positions géographiques signées, une complexité qui n'est pas très appréciable dans les réseaux de capteurs, connus pour avoir une faible capacité mémoire.

La méthode de sélection de témoins du protocole *LSM* est aléatoire. Cependant, selon les simulations réalisées dans le cadre des travaux présentés dans [CDPMM07], la répartition géographique des témoins dans le réseau n'est pas uniforme. En fait, on retrouve la moitié d'entre eux dans la zone la plus au centre (elle représente 20% de la surface totale du réseau) et seulement 1,75% de témoins dans la zone la plus externe du réseau (cette zone représente aussi 20% de la surface totale du réseau). Ce fait, influe négativement d'une part sur la répartition des coûts dans le réseau (ceci rend plus courte la durée de vie des nœuds les plus sollicités, ce qui peut perturber le fonctionnement du réseau) et d'autre part sur la résistance du protocole à la corruption de témoins dans la mesure où l'attaquant aura une connaissance plus précise des positions géographiques des futurs témoins.

Les auteurs de [CDPMM07] proposent également un protocole de détection qui, en plus de réduire les coûts (coût mémoire constant, coût des communications de l'ordre de $O(\sqrt{n})$), il assure une répartition équitable de ceux-ci sur l'ensemble des nœuds du réseau. Grâce à un nombre aléatoire généré par une tierce partie de confiance à chaque exécution d'un tour de détection, et d'une fonction de sélection de témoins publique et déterministe basée sur l'identité du nœud et sur ce nombre aléatoire, les témoins sont choisis de manière déterministe mais ne sont connus qu'au moment de la détection. Ceci permet d'avoir une complexité équivalente au protocole *multicast déterministe* de Parno et *al.* avec la probabilité qu'un nœud soit choisi comme témoin d'un nœud avoisinant $1/n$ (ce qui signifie que leur méthode de sélection de témoins ne divulgue aucune information sur les futurs témoins).

L'inconvénient de cette approche réside au niveau du calcul et de la distribution même du nombre aléatoire utilisé pour la sélection des témoins. Effectivement, la mise en place de ce nombre nécessite soit l'intervention d'une ou plusieurs stations de base, soit l'utilisation d'un protocole additionnel comme un protocole d'élection de leader qui est réputé pour être très sensible aux attaques par déni de service et qui nécessite des frais de communication supplémentaires.

Un autre protocole distribué plus vulnérable aux attaques contre le processus de sélection de témoins que le protocole *LSM* de Parno et *al.*, mais utilisant des quantités inférieures de mémoire que ce dernier a été proposé dans [ZAS+07]. Ce protocole améliore la sécurité du protocole *multicast déterministe* proposé dans [PPG05] sans augmenter considérablement les coûts. Pour cela, au lieu de sélectionner individuellement chaque témoin dans tout le réseau (ce qui peut occasionner un coût en communication proportionnel au nombre de témoins), il utilise une fonction de sélection de témoins publique et déterministe basée sur l'identité d'un nœud qui renvoie les coordonnés d'une zone géographique dans laquelle la déclaration est diffusée. Cette approche engendre des frais de communication supplémentaires de $O(s)$ et un coût en mémoire de $O(1)$ par nœud, s étant le nombre de témoins dans chaque zone. Cependant, cette approche divulgue l'emplacement des futurs témoins, ce qui la rend moins résistante aux corruptions de ceux-ci.

On peut remarquer que toutes ces solutions suivent une approche passive de détection. Autrement dit, les nœuds déclarent leur position géographique spontanément, et celles-ci seront vérifiées par une entité centrale ou de manière coopérative par les autres nœuds témoins du réseau.

Dans ce chapitre, nous présentons un protocole qui est un des principaux résultats de nos travaux de thèse. Bien que ce protocole donne lieu à la même complexité asymptotique que [CDPMM07], il a un coût légèrement supérieur à ce protocole en pratique. Son principal avantage par rapport à [CDPMM07] est sa robustesse (par rapport à [ZAS+07] il y a également moins de communications). Nous ne nous basons sur aucun protocole de choix commun d'un aléa (contrairement à [CDPMM07]), sujet à une attaque par déni de service, tout en ayant des témoins uniformément distribués (contrairement à [ZAS+07]). Ce protocole est présenté dans la section 4.4.

4.1 Réseaux de capteurs

4.1.1 Caractéristiques

Un réseau de capteurs sans fil [ASSC02] est un réseau multisaut constitué de minuscules dispositifs, généralement fixes, qui sont soit posés à un endroit précis, soit dispersés aléatoirement (par exemple déployés par voie aérienne à l'aide d'avions ou d'hélicoptères). À cet effet, les protocoles utilisés dans les réseaux de capteurs doivent posséder des algorithmes

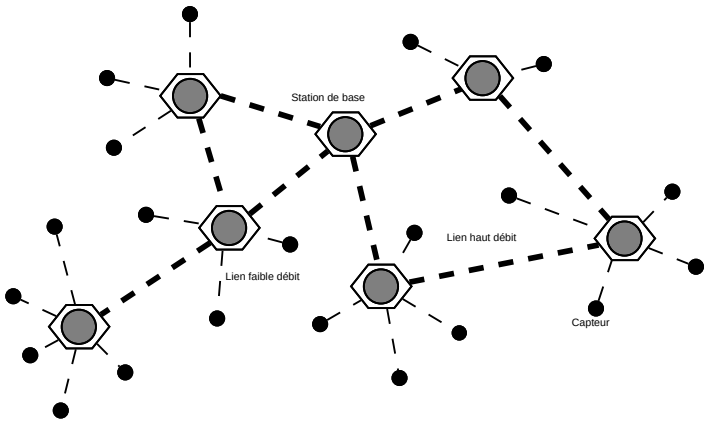

FIGURE 4.1 – Architecture d'un réseau de capteurs. Les capteurs utilisent des liens à faible portée et à faible débit alors que les stations de base utilisent des liens à longue portée et à haut débit.

d'auto-organisation. Afin de résister aux déploiements, ces capteurs doivent être très solides et de plus, ils doivent aussi pouvoir survivre dans les conditions les plus extrêmes dictées par leur environnement d'utilisation (p. ex., dans un feu ou dans l'eau).

Les réseaux de capteurs sont utilisés dans le but de récolter des informations sur un phénomène bien précis (p. ex., la température ou la pression de l'atmosphère) afin de les acheminer vers des centres de traitement. En effet, le besoin d'une surveillance continue d'un environnement donné est assez courant dans diverses activités de la société. Les processus industriels, les applications militaires, la surveillance médicale ainsi que l'agriculture de précision ne sont que quelques exemples de la panoplie d'applications possibles des réseaux de capteurs.

Le nombre de capteurs peut être très élevé, ce qui rend le coût de fabrication de ces dispositifs une question cruciale. Effectivement, les contraintes de prix, de performance (aucune application ne serait rentable si le rapport performance/prix était trop élevé) et de taille sont de plus en plus sévères, il est donc nécessaire d'allier performance, coût de fabrication et taille, ce qui explique de ce fait les capacités de calcul, de communication, de stockage et d'énergie très limitées des réseaux de capteurs existants aujourd'hui sur le marché. Un exemple de capteur est donné ci-dessous.

Exemple 6. *MICA mote [mico4] :*
Les capteurs ont un processeur de 4 *MHz* 8*-bit Atmel ATMEGA*103 *CPU avec* 128*KB de mémoire d'instructions,* 4 *KB de RAM et* 512*KB de mémoire flash. La CPU consomme* 5.5 *mA (à* 3*V) à l'état*

actif, et deux degrés de moins en mode veille. La fréquence radio est de 912*MHz avec un débit de* 40*Kbps et un rayon de propagation d'environ* 12 *mètres. L'interface radio consomme* 4.8*mA (à* 3*V) en mode réception, jusqu'à* 12*mA en mode transmission, et* 5 *µA en mode veille.*

Les réseaux de capteurs ont souvent un ou plusieurs points de contrôle centralisés appelés stations de base. Une station de base peut être une passerelle vers un autre réseau, un calculateur performant, un centre de stockage de données, ou un point d'accès pour un administrateur. Une station de base est utilisée afin de diffuser des requêtes de contrôle aux capteurs ou bien recevoir, périodiquement, des informations d'eux. Elles sont beaucoup plus performantes que les capteurs. Elles possèdent des processeurs, mémoires, autonomie et interface de communication équivalents aux stations de travail ou aux ordinateurs portables. La figure 4.1 donne une représentation simplifiée d'un réseau de capteurs.

4.1.2 Contraintes de conception

Un réseau de capteurs sans fil est un cas particulier de réseau ad hoc dans lequel plusieurs contraintes supplémentaires apparaissent. En raison de ces contraintes, il est difficile de recourir directement aux approches de sécurité existantes. Par conséquent, afin d'élaborer des mécanismes de sécurité efficaces, il est nécessaire de connaître et de comprendre ces contraintes. Nous n'allons pas recenser ici toutes les contraintes possibles, mais nous nous contenterons uniquement de celles ayant un lien direct avec la sécurité de ces réseaux.

Ressources matérielles limitées

Contrairement aux stations de base qui jouissent d'une capacité de communication importante, les capteurs sont contraints par des considérations d'économie d'énergie, de débit de transmission et de portée radio. De ce fait, les capteurs doivent utiliser une communication de proche en proche pour atteindre les stations de base les plus proches. La gestion de l'énergie est une question cruciale dans les réseaux de capteurs. Prenons l'exemple de MICA 2 : à plein régime, la durée de vie des batteries ne peut excéder deux semaines. Si l'on veut avoir des capteurs dont la durée de vie peut atteindre une année, une durée habituelle dans les réseaux de capteurs, ceux-ci doivent appliquer le principe du moindre effort en économisant d'une part l'activité processeur et d'autre part les communications sans fil. Il est important de noter que les transmissions radio sont très gourmandes en énergie. En effet, chaque bit transmis consomme une énergie équivalente à l'exécution d'environ 800 à 1000 instructions.

Les limitations en ressources physiques des réseaux de capteurs sont donc beaucoup plus importantes que dans les réseaux ad hoc classiques, ce qui impose de sérieux défis lors de la conception des protocoles dédiés aux réseaux de capteurs et tout particulièrement lors de la mise en œuvre de mesures liées à la sécurisation du réseau. En effet, les capteurs ont une capacité de calcul très limitée : de ce fait, la cryptographie à clés publiques a longtemps était impossible à appliquer et la cryptographie symétrique est à utiliser avec modération. Avec une mémoire RAM qui varie seulement entre 4 KB pour MICA 2 et 10 KB pour Telos B mote, la mémoire doit être aussi gérée de manière efficace. Enfin, vu que les communications engendrent une consommation d'énergie considérable, les protocoles de sécurisation du réseau doivent minimiser le nombre de messages échangés pour ne pas impacter significativement sur la durée de vie des capteurs.

Le manque de ressources matérielles et la faible puissance des capteurs rendent difficile l'utilisation des algorithmes à clés publiques traditionnels tels que le protocole d'échange de clés de Diffie-Hellman ou le cryptosystème RSA [RSA78]. Cependant, avec l'apparition des courbes elliptiques, l'application de ce type de systèmes redevient très envisageable et des implémentations de systèmes cryptographiques à clés publiques pratiques pour réseaux de capteurs ont ainsi été proposées [LN08]. Par exemple, TinyEcc permet la signature et la vérification d'une signature en moins de 100 millisecondes pour les réseaux de capteurs actuels, ce qui constitue une hypothèse raisonnable pour un nombre limité de signatures et de vérifications de signatures.

Vulnérabilité physique

Les réseaux de capteurs sont des réseaux auto-organisés, qui une fois déployés sont censés fonctionner de manière autonome et sans intervention humaine. En outre, les contraintes liées aux coûts de fabrication, aux ressources matérielles, rendent difficile la mise en place d'une protection physique ou logicielle. Ainsi, les capteurs sont exposés aux captures, aux falsifications, aux vols de données et aux altérations physiques par un adversaire. Dans le pire des cas, un adversaire peut être en mesure de prendre de manière non détectable le contrôle d'un capteur et de compromettre ses clés cryptographiques.

On appelle attaque physique [WGYX05, WGS+09], toute attaque menée sur le capteur en tant que composant matériel électronique. Ces attaques supposent une connaissance approfondie de l'architecture matérielle du capteur. Les attaques physiques contre les capteurs peuvent être classifiées en deux catégories :

- attaques invasives : cette attaque utilise des techniques de sondage qui nécessitent des accès aux composants électroniques de bas niveau du capteur ;

- attaques non invasives : ces attaques exploitent les canaux cachés du microprocesseur. Dans ce type d'attaques, on va mesurer un paramètre physique extérieur de la puce du capteur pendant son activité ; on ne touche donc pas à l'intégrité physique de la puce. Ce paramètre physique peut être le temps de calcul, le courant, ou le champ électromagnétique émis par la puce.

Une étude réalisée en 2005 [HBH+05] montre comment des capteurs standards, tels que les capteurs MICA2, peuvent être compromis en moins d'une minute.

Problème d'amorçage

Si un réseau de capteurs est déployé via un positionnement aléatoire des capteurs (par exemple par le biais d'un avion), les protocoles opérant sur les capteurs sont incapables de prévoir quels nœuds sont à portée radio les uns des autres après le déploiement. Même si les nœuds sont déployés manuellement, il est coûteux de déterminer à l'avance l'emplacement de chaque nœud en raison du grand nombre de capteurs qui sont souvent impliqués dans ce type de réseaux. De ce fait, les protocoles de manière générale et en particulier les protocoles de sécurité destinés aux réseaux de capteurs ne doivent pas supposer la connaissance préalable des positions des nœuds ou bien qui sera le voisin de qui dans le réseau.

Ainsi, la phase d'initialisation ou d'amorçage des réseaux de capteurs doit satisfaire les exigences suivantes :

- les capteurs doivent être capables d'établir des communications sécurisées entre chaque paire de nœuds ;
- les protocoles doivent être auto-organisés (c'est-à-dire, ne nécessitent pas les services d'une station de base) ;
- Les nœuds légitimes qui intègrent le réseau, après le déploiement de celui-ci, doivent être capables d'établir des connexions sécurisées avec les nœuds déjà déployés. Cela implique que les informations d'amorçage doivent toujours être présentes dans le réseau et ne peuvent donc pas être simplement effacées après le déploiement afin d'empêcher par exemple la corruption des nœuds en cas de capture ;
- Les nœuds non autorisés ne doivent pas être en mesure d'établir des communications avec les nœuds légitimes du réseau et d'obtenir ainsi un accès au réseau.

4.2 *Cadre de travail*

4.2.1 Notations

Nous utilisons dans ce chapitre les notations suivantes, reprises de l'article [PPG05] de Parno et *al.*

n	Le nombre de nœuds dans le réseau
d	Degré moyen d'un nœud
p	La probabilité pour un voisin de retransmettre les paquets reçus
g	Le nombre de témoins sélectionnés par chaque voisin
l_α	La position géographique que le nœud α déclare occuper
$H(M)$	Le Hash de M
K_α	La clé publique du nœud α
K_α^{-1}	La clé privée du nœud α
$\{M\}_{K_\alpha^{-1}}$	La signature de M calculée par le nœud α
S	L'ensemble de tous les identifiants possibles

4.2.2 Objectifs

Le défi majeur d'un protocole de détection de répliques est, d'une part, d'obtenir un taux de détection très élevé et d'autre part de minimiser le coût en termes de communication et d'occupation mémoire des capteurs, tout en étant résistant aux attaques contre le processus de détection (en particulier contre le processus de sélection de témoins).

Notre premier objectif est de détecter avec une très forte probabilité la présence de nœuds répliqués dans un réseau de capteurs en utilisant une approche de détection distribuée. Notre protocole doit être capable de révoquer les nœuds répliqués de telle manière à ce que tous les autres nœuds du réseau arrêtent de communiquer avec eux.

Notre deuxième objectif est de minimiser le nombre de messages que chaque nœud va émettre. Pour cela, nous proposons un protocole avec une probabilité de détection exponentiellement proche (en un paramètre k) de 100% avec un nombre de messages émis par nœud de l'ordre de $k\sqrt{n}$.

Notre troisième objectif est de construire un protocole qui nécessite une quantité constante et infime de mémoire.

Enfin, nous voulons que la méthode de sélection de témoins adoptée soit résistante au modèle d'attaquant que nous présentons dans la section 4.2.4.

4.2.3 Caractéristiques du réseau de communication

Certaines hypothèses sont habituelles dans le contexte de la détection de répliques dans les réseaux de capteurs. Nous les reprenons ici.

Nous considérons que les nœuds sont fixes et qu'un système permettant à chaque nœud d'obtenir ses coordonnées géographiques existe [NS03, DPG01, CJ07].

Chaque nœud est supposé posséder un identifiant unique $ID \in S$. Nous supposons aussi l'existence d'un système de gestion de clés publiques basé sur les identités des nœuds de telle sorte que chaque nœud ayant une identité α puisse obtenir, en utilisant une fonction f connue de tous, la clé publique K_α qui lui est associée. Chaque nœud possède aussi une clé privée associée à sa clé publique (elle est par exemple intégrée au capteur lors de sa fabrication).

Afin de faciliter l'analyse des performances, on suppose que le diamètre du réseau est approximativement de l'ordre de $O(\sqrt{n})$. Si ce n'est pas le cas, l'analyse de performance doit être adaptée en conséquence pour la nouvelle topologie réseau. Dans de tels modèles, deux nœuds pris au hasard sont séparés par environ $\sqrt{n}/2$ sauts. Nous supposons donc que l'envoi d'un message d'un nœud à un autre génère $O(\sqrt{n})$ messages.

4.2.4 Modèle d'attaquant

Un attaquant peut capturer, corrompre, dupliquer et déplacer de manière adaptative un ensemble fixe de capteurs, mais il est incapable de créer une nouvelle identité, car elle doit être liée à une paire de clés publiques/privées établie à l'initialisation du réseau.

Les répliques contrôlées par un attaquant peuvent communiquer et collaborer. Nous supposons aussi que chaque nœud répliqué essaye de rester le plus discret possible et que des attaques actives sont facilement détectables par un protocole de balayage (Parno et *al.* proposent Swatt [SPvDK04]) ou par l'intervention de l'homme. Ceci signifie qu'un attaquant ne peut pas intercepter ou perturber un nombre significatif de communications. En d'autres termes, les nœuds peuvent refuser de jouer le jeu, tricher, mais ne peuvent pas empêcher l'exécution du protocole.

Afin de résister à la détection, l'attaquant peut aussi corrompre un ensemble fixe de capteurs qui peuvent potentiellement faire partie de l'ensemble des futurs témoins. Pour ce faire, l'attaquant peut choisir l'ensemble des nœuds à corrompre dans tout le réseau ou bien à l'intérieur d'une zone géographique bien délimitée. Il peut choisir ces nœuds d'une manière complètement aléatoire (sans connaitre les détails d'implémentation du protocole

de détection mis en place dans le réseau) ou bien d'une manière ciblée afin d'accroitre les chances de réussite de son attaque (la réplication d'un ou plusieurs nœuds). Par exemple, l'attaquant choisit les nœuds ayant les plus grandes probabilités d'être sélectionnés comme témoins.

En effet, un des principaux problèmes lors de la conception d'un protocole de détection de l'attaque par réplication de nœuds est la sélection de l'ensemble des témoins. Si l'attaquant peut connaître les futurs témoins avant l'exécution du protocole de détection, il peut les corrompre afin de rendre son attaque indétectable. L'attaquant peut utiliser une multitude d'informations afin de prévoir la probabilité qu'un nœud donné soit un futur témoin. Dans [CDPMM07], les auteurs identifient deux types de prédiction :

- prédictions basées sur les identités ;
- prédictions basées sur les positions géographiques.

On dit d'un protocole de détection qu'il est imprévisible dans le choix des identités des futurs témoins s'il n'offre aucune information sur ces identités ni aux entités internes, ni aux entités externes au réseau. De la même manière, le processus de sélection de témoins est dit indépendant des positions géographiques, si la probabilité qu'un nœud soit un témoin potentiel ne dépend pas de sa position géographique dans le réseau.

4.3 *Travaux existants*

L'article fondateur du domaine est publié par Parno et *al.* en 2005 [PPG05] et contient un ensemble de propositions plus ou moins triviales pour résoudre le problème de la détection de répliques, suivi de leur proposition principale *Line Selected Multicast*. Nous commençons cette section par une énumération des propositions de Parno et *al.* puis nous présentons deux autres travaux postérieurs, un de Conti et *al.* [CDPMM07], et un de Zhu et *al.* [ZAS+07].

4.3.1 Protocoles de Parno et *al.*

Approche centralisée

Dans une approche centralisée, chaque nœud envoie à la station de base la liste de ses voisins avec leurs positions géographiques. La station de base peut ainsi vérifier s'il n'existe aucun nœud se trouvant au même moment dans des positions différentes (non adjacentes). Si elle découvre une ou plusieurs copies d'un nœud, elle peut les révoquer en diffusant un message de révocation authentifié dans tout le réseau. La station de base reçoit et traite $O(n)$

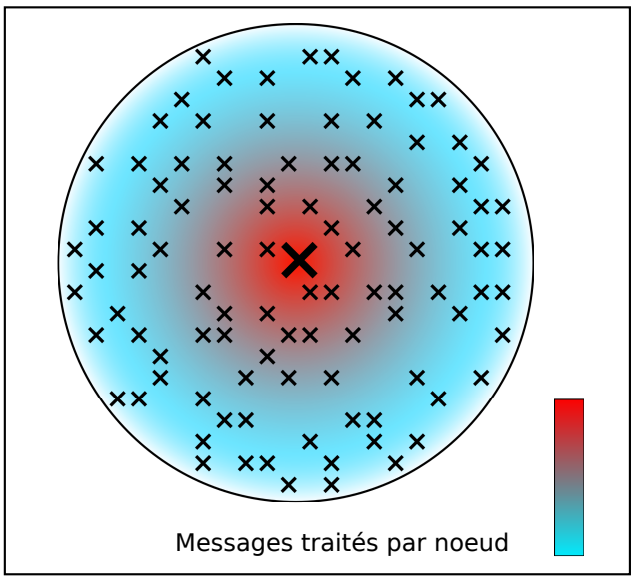

FIGURE 4.2 – Concentration de messages dans l'approche centralisée

messages, mais ceci n'est généralement pas un problème, car les stations de base n'ont pas les contraintes de mémoire, de communication et de calcul des capteurs. Cette approche génère un total de $O(n\sqrt{n})$ messages et l'usage de la mémoire est quasi nul.

Le premier problème qu'on peut remarquer dans cette approche est le fait qu'un nœud peut facilement tricher et prétendre être un voisin légitime d'un autre nœud alors que ce n'est pas le cas. Ce problème peut être facilement résolu en utilisant un protocole fonctionnant en deux temps. Dans un premier temps, chaque nœud diffuse localement (à ses voisins) ses coordonnées géographiques signées. Dans un second temps, chaque nœud envoie les d déclarations signées reçues de ses voisins à la station de base. Cette solution permet de résoudre le problème présenté précédemment et il permet aussi de forcer les nœuds à diffuser leurs positions géographiques. En effet, si un nœud ne diffuse pas localement ses coordonnées géographiques, ses voisins ne coopéreront pas avec lui, ce qui va l'exclure de l'activité du réseau.

Les deux principaux inconvénients de cette approche sont l'existence d'un seul point de vulnérabilité et la nécessité pour un réseau d'avoir une station de base puissante et accessible à tout moment. Ces problèmes peuvent être évités en mettant en place des solutions distribuées. Enfin, il est important de remarquer un autre inconvénient majeur : le traitement des messages à travers le réseau n'est pas équilibré. Effectivement, comme tout

le trafic engendré par cette approche centralisée est envoyé vers la même destination à savoir la station de base, les capteurs qui sont proches de la station de base seront les plus sollicités. Le nombre de messages reçus et envoyés par les nœuds au voisinage de la station de base est de l'ordre de $O(n)$ alors que ceux qui sont à la périphérie de la station de base n'envoient que $O(1)$ massages (voir figure **4.2**).

Détection locale

Cette méthode ne résout que partiellement le problème de détection de nœuds répliqués. Elle permet en effet d'éviter l'utilisation d'une station de base centrale, en s'appuyant sur la coopération entre les voisins d'un nœud pour réaliser la détection. En utilisant un mécanisme de vote, les voisins vont tenter de parvenir à un consensus concernant la légitimité d'un nœud donné. Malheureusement, cette méthode est incapable de détecter les répliques se trouvant dans deux voisinages disjoints du réseau (c'est-à-dire, les répliques se trouvant au minimum à deux sauts l'une de l'autre). Ces schémas se présentent souvent comme un mécanisme de détection intégré dans un système de distribution de clés. Par exemple, dans [CPS03], leur système de pré-distribution de clés permet à chaque nœud d'authentifier son voisin grâce à une clé unique délivrée à chaque couple de nœuds. Ils proposent aussi que ce processus d'authentification se fasse de manière coopérative entre les nœuds du même voisinage.

Diffusion nœud-réseau

Une approche relativement simple consiste à ce que chaque nœud diffuse dans tout le réseau une déclaration signée de sa position géographique et sauvegarde les déclarations de ses d voisins immédiats. Si un nœud reçoit une déclaration signée d'un nœud qui est en conflit avec celle de l'un de ses voisins, il diffuse dans tout le réseau une preuve de révocation contenant les déclarations en conflit. En supposant qu'il existe un mécanisme efficace de diffusion dans le réseau, chaque diffusion va nécessiter $O(n)$ messages, et donc le coût total des communications du protocole sera de $O(n^2)$. De plus, chaque nœud doit stocker les déclarations de ses voisins, par conséquent la complexité en termes de mémoire au niveau de chaque nœud est $O(d)$. Afin que la détection soit assurée, chaque nœud doit obligatoirement diffuser ses coordonnées géographiques signées, cet objectif est facilement atteint en mettant en place un mécanisme de sanction pour tout nœud refusant de participer au protocole (les voisins d'un nœud refuseront de coopérer avec tout voisin qui ne diffuserait pas sa position géographique). En supposant que les diffusions atteignent tous les nœuds du réseau, ce

schéma assure un pourcentage de détection de 100%.

Multicast déterministe

Parno et *al.* proposent également dans [PPG05] un autre protocole relativement simple : *le multicast déterministe*. Afin d'améliorer les coûts en termes de communication, ce protocole de détection permet le partage des déclarations de position uniquement avec un sous ensemble limité de nœuds témoins choisis de manière déterministe. Il s'appuie sur une fonction publique F qui, pour chaque nœud ayant une identité α, produit un ensemble de nœuds témoins $F(\alpha)$. Chaque nœud α effectue une diffusion locale de sa position géographique signée et chaque voisin la retransmet avec une probabilité p à un sous-ensemble de témoins du nœud qui sont choisis aléatoirement parmi les $F(\alpha)$ témoins. Les témoins sont choisis en fonction des identifiants des nœuds. Si un attaquant duplique un nœud, les témoins recevront deux positions géographiques différentes avec une même identité. Ce conflit de positions va constituer une preuve suffisante pour lancer une opération de révocation des nœuds répliqués.

La probabilité de faire suivre une déclaration, la taille du sous-ensemble de témoins choisi, et la taille de l'ensemble de témoins retourné par la fonction F sont des paramètres du protocole. Parno et *al.* proposent dans [PPG05] que chaque voisin fasse suivre la déclaration à $\frac{g \times ln(g)}{d}$. D'après le problème de la collecte de coupons [CLR90] (collector's problem) ceci permet à chacun des témoins de recevoir au moins une déclaration, ce qui conduit à $O(g \times ln(g) \times n\sqrt{n})$ messages envoyés par nœud.

Le problème majeur de cette approche est que la fonction F est connue de tous. Un attaquant a juste besoin de capturer le nœud α et ses témoins $F(\alpha)$ pour être capable de dupliquer le nœud α autant de fois qu'il le veut. Il est possible de choisir F de manière à ce que le nombre de témoins g résultant de cette fonction soit très grand, mais comme le coût des communications est de $O(g \times ln(g) \times n\sqrt{n})$. Ceci aura une forte répercussion sur la durée de vie des capteurs.

Protocole multicast stochastique (Randomized multicast protocol)

Dans ce protocole, la fonction F utilisée dans le protocole de détection *multicast déterministe* est remplacée par une sélection aléatoire. Chaque nœud fait suivre les déclarations de position reçues avec une probabilité p vers un ensemble de g témoins choisis au hasard. Pour $p \times d \times g \simeq \sqrt{n}$, chaque nœud aura donc $O(\sqrt{n})$ témoins. Le paradoxe des anniversaires (the birthday paradox) assure que si deux répliques d'un nœud envoient deux déclarations de

position signées différentes, il y a une très forte probabilité qu'un même témoin reçoive les deux déclarations en conflit.

L'inconvénient majeur d'un tel protocole est comme l'envoi d'un message coûte $O(\sqrt{n})$ messages à chaque nœud, contacter $\sqrt{n}$ témoins par nœud coûtera par conséquent $O(n)$ messages. Le trafic total généré est donc de $O(n^2)$, un coût équivalent à celui de l'approche simpliste de *diffusion nœud-réseau*. Cependant, ce protocole est considéré par Parno et *al.* comme le point de départ de la construction d'un protocole plus efficace en termes de communication, à savoir le protocole *line-selected multicast*.

Line-Selected Multicast protocol(LSM)

L'idée principale de ce protocole est de choisir les témoins de telle manière à ce qu'un attaquant ait des difficultés à tous les contrôler, tout en faisant attention aux coûts engendrés pour atteindre ces témoins. La difficulté de cette approche réside dans le choix de la méthode qui permette d'obtenir une très forte probabilité d'intersection entre les ensembles de témoins de deux répliques d'un nœud.

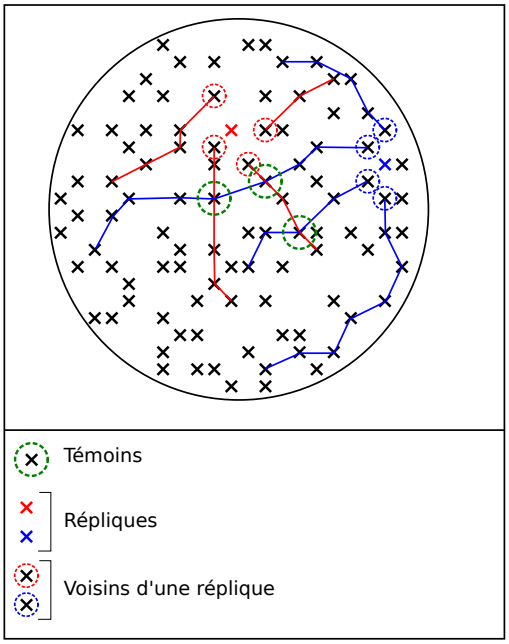

FIGURE 4.3 – Intersection des ensembles de témoins dans le cas du protocole line selected multicast

Parno et *al.* proposent de sélectionner, comme témoins, les nœuds intermédiaires vers une

destination qu'il choisit aléatoirement. Cette démarche a deux avantages très intéressants. En premier, des lignes presque droites sont créées dans le réseau avec une forte probabilité pour qu'elles se croisent les unes avec les autres. Le second avantage est qu'avec cette méthode $O(\sqrt{n})$ témoins seront choisis pour vérifier la déclaration de position d'un nœud, mais avec un coût en communication inférieur à $O(\sqrt{n} \times \sqrt{n})$, le coût obtenu dans le cas d'une sélection complètement aléatoire des témoins.

Chaque nœud diffuse localement sa déclaration de position signée. Ensuite, chacun de ses d voisins décide avec une probabilité p de faire suivre la déclaration reçue. Un voisin qui décide de retransmettre la déclaration, choisit au hasard un nœud destination et lui envoie la déclaration. Tous les nœuds intermédiaires qui feront suivre la déclaration vont la sauvegarder et ainsi créer une ligne de nœuds témoins en direction du nœud destination. Parno et *al.* donnent dans leur article quelques résultats théoriques intéressants qui montrent que, dans le cas idéal, le nombre de lignes nécessaires, afin que deux répliques aient une très forte probabilité pour que leurs lignes se croisent, est petit. Ils affirment qu'en prenant la valeur de cinq lignes par nœud la probabilité d'intersection est de 95%.

En prenant $p = K/d$ on assure à chaque nœud d'avoir environ K lignes de témoins. Pour une telle valeur, le coût total en communication sera de $O(K \times n\sqrt{n})$. La probabilité d'intersection entre les ensembles de témoins de deux répliques éventuelles, suivant l'augmentant de la valeur de K, avoisinerait de manière exponentielle la valeur de 1. Chaque nœud aura à sauvegarder $O(K \times \sqrt{n})$ déclarations de position

4.3.2 Le protocole RED

Le protocole *Randomized Efficient Distributed* (*RED*) a été proposé dans [CDPMM07] pour la détection passive de répliques dans un réseau de capteurs.

En termes de sécurité, leur protocole se veut, d'une part, plus efficace en termes de détection de nœuds répliqués que le protocole *LSM* et d'autre part, résistant aux attaques contre le processus de sélection de témoins, que ce soit contre des prédictions basées sur les identités ou bien sur des prédictions basées sur les positions géographiques.

En termes d'utilisation des ressources matérielles, en plus d'offrir une utilisation constante de la mémoire des capteurs et un coût de communication moyen de $O(\sqrt{n})$, leur méthode de détection répartit ces coûts uniformément entre tous les nœuds du réseau. En effet, d'après les résultats d'une simulation réalisée dans leur article, comparant leur protocole avec le protocole *LSM* de Parno et *al.* (cf. figure 4.4), ils montrent que, dans le protocole LSM, la zone centrale du réseau (qui représente 20% de la surface globale du réseau) regroupe plus de 50% des témoins, alors que la zone la plus externe (à la périphérie

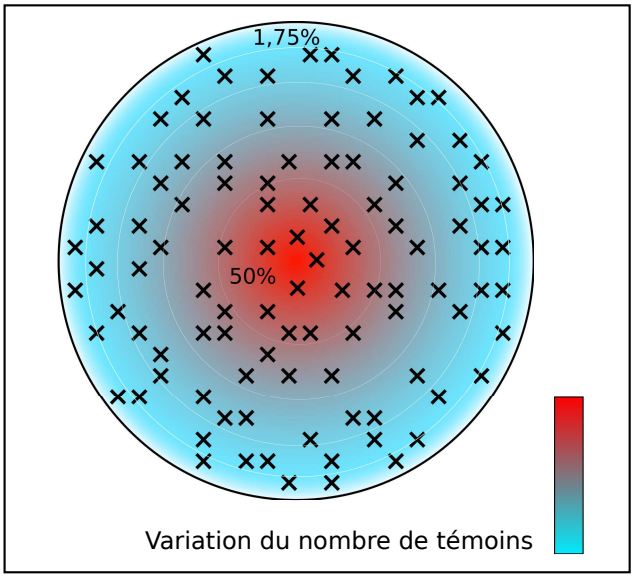

FIGURE 4.4 – Répartition des coûts et des témoins dans le protocole *LSM*

du réseau), qui correspond aussi à 20% de la surface globale du réseau, regroupe seulement 1.75% par rapport à l'ensemble de tous les témoins.

Ce résultat expérimental est aussi très intéressant dans l'analyse de la résistance des deux protocoles aux attaques contre le processus de sélection des futurs témoins. On peut remarquer, d'après ces résultats, que le protocole LSM ne vérifie pas la propriété de non prédiction basée sur les positions géographiques, car la probabilité qu'un nœud du centre du réseau soit sélectionné comme témoin est plus grande que la probabilité de sélectionner un témoin dans la périphérie du réseau. Par contre, dans le protocole RED, le choix des témoins est fait de manière équiprobable sur toute la surface du réseau, ce qui ne révèle aucune information sur les positions géographiques des futurs témoins.

L'idée du protocole RED est d'utiliser une approche de détection déterministe, mais dont le résultat n'est connu qu'à la dernière minute, au début de la phase de détection. Pour ce faire, chaque exécution du protocole se fait en deux étapes. Durant la première étape, une valeur aléatoire (*Random*) est partagée entre tous les nœuds du réseau. Ceci peut être réalisé à l'aide d'une entité centrale (p. ex., diffusion de cette valeur par une station de base) ou bien en utilisant des algorithmes distribués d'élection de leader. Durant la deuxième étape (la phase de détection), chaque nœud signe et diffuse localement sa déclaration de position (son identité et sa position géographique). Pour chaque nœud, chacun de ses voisins envoie

(avec une probabilité p) la déclaration reçue à $g >= 1$ nœuds (définis par leurs positions géographiques) témoins pseudo-aléatoirement choisis. La fonction de génération pseudo-aléatoire prend comme argument : l'identifiant du nœud, la valeur aléatoire *Random* et le nombre g de témoins nécessaires à la détection. La déclaration est reçue par tous les g témoins proches des positions aléatoirement tirées. L'utilisation des positions géographiques à la place d'utiliser directement les identifiants des témoins est motivée, selon les auteurs, par le fait qu'en procédant ainsi on tiendrait mieux en considération les arrivées et les départs des capteurs dans le réseau.

Pour chaque déclaration reçue, un témoin éventuel :

- vérifie la signature reçue ;
- vérifie la fraîcheur du message reçu ;

Pour tout message ayant passé ces deux tests avec succès, chaque nœud témoin vérifie, tout d'abord, s'il a déjà reçu une déclaration comportant le même identifiant. Si le test se révèle négatif, le nœud témoin sauvegarde la déclaration reçue, dans le cas contraire, il vérifie si elles sont émises à partir du même emplacement géographique. Si le test est négatif, le témoin lance une procédure de révocation vis-à-vis du nœud déclarant.

Cette approche est très intéressante et donne des résultats très satisfaisants surtout en termes d'occupation mémoire (puisqu'elle est constante) et de résistance aux attaques contre le processus de sélection de témoins (puisqu'ils sont uniformément choisis). Cependant, elle repose sur une hypothèse de départ qui est assez forte dans la mesure où ils supposent la mise en place d'un nombre aléatoire qui est déterminé juste avant un round d'exécution, en utilisant : soit une infrastructure lourde (p. ex., plusieurs stations de base, un satellite, etc.) ; soit un algorithme d'élection de leader (par exemple [CBS05]).

L'utilisation d'une infrastructure, que ce soit sous la forme d'une série de bases, ou d'un satellite est une très mauvaise propriété pour un protocole dans le contexte des réseaux de capteurs, puisqu'elle pervertit la nature auto-organisée et indépendante de ces réseaux. Quant à l'utilisation d'un protocole de sélection de leader, nous avons également une opinion mitigée. En effet, le protocole proposé par les auteurs de [CBS05], est un protocole d'élection locale qui permet de choisir une série de leaders (un peu comme les MPR dans OLSR) et pas un seul leader. Cette élection est réalisée dans [CBS05] dans le but de réduire les coûts d'une diffusion dans le réseau. Or, dans les travaux sur la réplication on suppose déjà qu'un protocole de diffusion efficace existe (et donc que le coût d'une diffusion est en $O(n)$). Il semble complexe de passer du protocole proposé dans [CBS05] vers un protocole de choix

d'un aléa sans que ce protocole soit très sensible aux attaques malveillantes (que ce soit pour biaiser le résultat ou empêcher l'exécution). Notre opinion est qu'un tel protocole sera donc fragile ou extrêmement couteux (en transformant par exemple un protocole de vote sécurisé).

4.3.3 Localized Multicast Approach (*LMA*)

Ce protocole, présenté dans [ZAS+07], est, tout comme *RED*, basé sur le principe de fonctionnement du protocole *multicast déterministe*. Lors de l'étude de ce dernier, nous avons pu constater que plus le nombre de témoins est grand, plus la résistance du protocole aux corruptions de témoins augmente, mais que ceci se fait malheureusement aux dépens des performances du réseau.

L'objectif du protocole LMA est de réduire ces coûts tout en ayant un grand nombre de témoins possibles. Pour ce faire, il utilise toujours une fonction de sélection de témoins déterministe, mais au lieu qu'elle renvoie les identifiants des futurs témoins, elle renvoie les coordonnées d'une ou plusieurs régions géographiques (appelées *cellules*), dans lesquelles les déclarations de position sont diffusées.

A cet effet, ils proposent deux versions du protocole : *Single Deterministic Cell (SDC)* et *Parallel Multiple Probabilistic Cells (P-MPC)*. Cette deuxième version du protocole est tout simplement une généralisation de la première (l'identité d'un nœud est associée à plusieurs cellules au lieu d'une seule dans le cas de la première version) afin d'augmenter la résistance du protocole contre la corruption des nœuds par un attaquant.

La zone de déploiement du réseau est représentée par une grille géographique composée de $a \times b$ cellules. Une cellule à la $i^{\text{ème}}$ ligne et à la $j^{\text{ème}}$ colonne est identifiée de manière unique par $c = i.b + j$. Une fonction de hachage est utilisée afin d'associer chaque identité L à une cellule de la grille. Autrement dit, en utilisant une fonction de hachage à sens unique $H()$, un nœud L est lié à une cellule D, avec $D = [H(ID_L) mod (a.b)] + 1$. Le format d'une déclaration de position se présente de la manière suivante : $[ID_L, l_L, SIG_{SK_L}(H(ID_L||l_L))]$ (on note par $||$ l'opération de concaténation).

A chaque round d'exécution du protocole, chaque nœud L diffuse localement sa déclaration de position. Chacun de ses voisins vérifie tout d'abord si la position déclarée l_L est plausible ou pas (par exemple, en se basant sur son emplacement géographique et sur le rayon de transmission d'un capteur) et la validité de la signature contenue dans la déclaration. Ensuite, il décide avec une probabilité p_f de faire suivre la déclaration reçue. Si un voisin décide de faire suivre la déclaration de position, il exécute la fonction de hachage afin de déterminer la cellule de destination D et envoie la déclaration.

Une fois que la déclaration arrive à D, le capteur qui la reçoit vérifie la validité de la signature et vérifie aussi que la cellule D correspond effectivement à l'identité contenue dans la déclaration. Dans le cas où les tests sont concluants, il diffuse la déclaration à l'intérieur de la cellule. Chaque nœud recevant la déclaration décide avec une probabilité p_s de la sauvegarder. Si un nœud ayant sauvegardé une déclaration reçoit une deuxième contradictoire, il fait suivre les deux déclarations à la station de base qui diffuse à son tour, dans tout le réseau, un message de révocation des répliques.

La sécurité de ce protocole dépend du nombre de nœuds s dans chaque cellule et son coût en communication est de $O(\sqrt{n}) + O(s)$. Ces coûts sont supérieurs à ceux du protocole *multicast déterministe* et sont toujours en fonction du nombre de témoins sélectionnés, mais offre une meilleure sécurité.

Par ailleurs, cette première version est meilleure, en termes d'occupation mémoire, que le protocole LSM de Parno et *al.*, mais s'avère moins résistante aux attaques contre le processus de sélection de témoins. En effet, alors que dans le protocole LSM les témoins sont choisis aléatoirement en considérant toute la surface du réseau, dans le cas de ce protocole, les témoins sont choisis à l'intérieure d'une petite zone géographique connue de tous. Ce qui va à l'encontre de la propriété de non prédiction basée sur les positions géographiques.

En exploitant ces informations sur l'emplacement géographique des futurs témoins, l'attaquant peut réduire très fortement le nombre de capteurs à corrompre ou à détruire pour assurer que ses répliques ne soient pas détectées. Si s est grand, corrompre tous les nœuds peut être coûteux, une destruction des capteurs dans la cellule étant une option plus simple. Enfin, il est intéressant de remarquer qu'à cause de la nature géographique du protocole, il n'y a pas vraiment besoin de détruire les capteurs de la cellule, il suffit de les déplacer hors de celle-ci pour qu'il n'y ai plus de témoins.

Afin d'augmenter la résistance aux attaques de ce protocole, ils ont proposé une deuxième version. Dans cette deuxième version, chaque identité peut être associée à plusieurs cellules en même temps. Cette deuxième version est plus sûre mais aussi plus coûteuse. Cependant, l'attaquant connaît toujours l'emplacement exact des nœuds à corrompre dont le nombre sera très inférieur à n (à moins que la diffusion soit globale au quel cas les coûts en communication seront trop importants).

4.4 Notre proposition : Protocole de détection actif

Notre proposition donne lieu à deux contributions. En premier lieu, nous proposons un nouveau type d'approches à savoir la détection active de nœuds répliqués. Cette approche

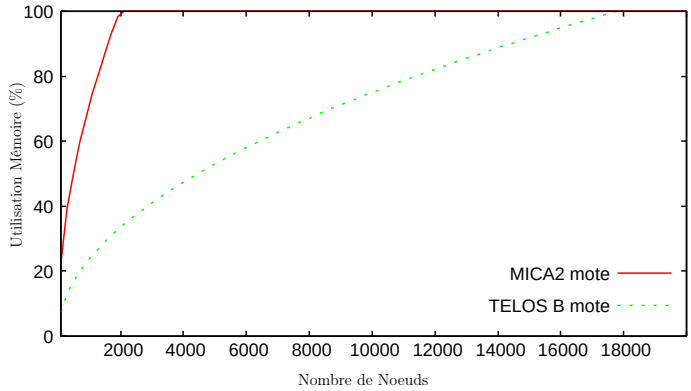

FIGURE 4.5 – Utilisation de la mémoire dans le protocole de Parno et *al.*

est une alternative à celle proposée par les autres protocoles et peut donner lieu à une famille de nouveaux protocoles dont le nôtre n'est qu'une instance. En deuxième lieu, le protocole proposé conserve les performances en mémoire et communication des protocoles *RED* et *LMA* sans avoir recours à un protocole d'élection de leader ni réduire l'espace des témoins possibles.

Plus précisément, dans notre protocole, chaque nœud teste de manière dynamique et périodique si certains nœuds du réseau, choisis aléatoirement, sont répliqués ou non. Pour cela, il utilise un certain nombre de relais aléatoirement situés dans le réseau afin de récupérer les positions géographiques du nœud testé. Nous démontrons que si un nombre suffisant de relais est choisi et que le nœud testé est répliqué, il est très improbable que tous les relais communiquent avec la même réplique. Le nœud testeur pourra donc détecter, à travers ces relais, que plusieurs répliques d'un nœud existent et pourra ainsi révoquer le nœud répliqué en utilisant comme preuve les positions géographiques en conflit.

Parno et *al.* soulignent dans leur papier que la détection passive n'est pas en soi un protocole, mais une famille de protocoles. Suivant la manière de choisir les nœuds témoins, plusieurs variantes de cette approche sont obtenues, chacune d'elles donnant des résultats différents en termes de performance. Ceci est valable aussi dans le cas de notre approche active de détection. En effet, suivant la manière dont sont testés les nœuds, plusieurs protocoles avec des performances variées peuvent être déduits.

Il faut remarquer que l'amélioration du coût en mémoire de *RED*, *LMA* et de notre protocole par rapport au protocole de Parno et *al.* va au delà des considérations asymptotiques et a un réel impact en pratique. En effet, pour être un peu plus précis,

pour le protocole de Parno et *al.*, chaque nœud sauvegarde en moyenne $3\sqrt{n}$ positions géographiques signées (avec les paramètres qu'ils proposent). Supposons qu'une position peut être encodée avec juste 20 bits (10 bits pour chaque coordonnée) et que la signature a une taille de 160 bits, alors la quantité de mémoire utilisée sera en moyenne de $540\sqrt{n}$ bits au niveau de chaque nœud du réseau. Parno et *al.* utilisent deux types de capteurs afin d'évaluer les performances de leur protocole, MICA 2 (avec 4 Ko de RAM) et le nouveau Telos B mote (avec 10 Ko de RAM). Ils supposent aussi l'utilisation du schéma de signature TinyECC [1], qui utilise 1 Ko de mémoire RAM pour sauvegarder l'empreinte générée. Ce qui laisse, pour la sauvegarde des déclarations signées reçues, 3 et 9 Ko respectivement pour les capteurs de type MICA 2 et Telos motes. La Figure 4.5 montre le pourcentage de mémoire utilisé lors de la sauvegarde des positions géographiques signées au niveau de chaque nœud suivant l'augmentation du nombre de nœuds dans le réseau. Nous constatons que MICA 2 ne peut pas gérer ce protocole pour un réseau de plus de 2000 nœuds, et que les capteurs Telos se limitent à un réseau de 18000 nœuds[2]. Cette observation est très significative dans la mesure où les réseaux de capteurs sont des réseaux où l'on considère que le nombre de nœuds peut atteindre 100000 nœuds, ce qui rend l'utilisation mémoire une question cruciale pour ce protocole.

4.4.1 Description

On ne construit pas dans notre protocole une base de données distribuée contenant les différentes déclarations de position des différents nœuds du réseau, ce qui permet de réduire fortement l'utilisation mémoire. L'idée est que chaque nœud, qu'on appelle ici nœud demandeur, vérificateur ou bien testeur, teste de manière autonome si k_1 autres nœuds du réseau, qu'il choisit aléatoirement, ont été répliqués ou pas. On les appelle les nœuds scrutés ou testés. Afin de tester si un nœud α est répliqué, k_2 nœuds du réseau seront choisis de manière aléatoire en leurs demandant de faire suivre au nœud α la requête demandant à α sa déclaration de position signée. Si deux répliques d'un nœud existent, elles recevront probablement une copie de la requête, et si leurs réponses sont contradictoires (chacune déclarant une position géographique différente de l'autre), le nœud vérificateur procédera à leur révocation du réseau. Nous allons prouver que la probabilité pour qu'un nœud répliqué soit détecté est exponentiellement proche de 1 en $min(k_1, k_2)$.

Le schéma que nous proposons est complètement distribué. Chaque nœud α exécute

[1] http ://discovery.csc.ncsu.edu/software/TinyECC/

[2] Ces bornes supposent que toute la mémoire non utilisée par TinyECC est disponible et utilisable que pour cette opération, ce qui ne sera probablement pas le cas, réduisant encore le nombre de nœuds maximal.

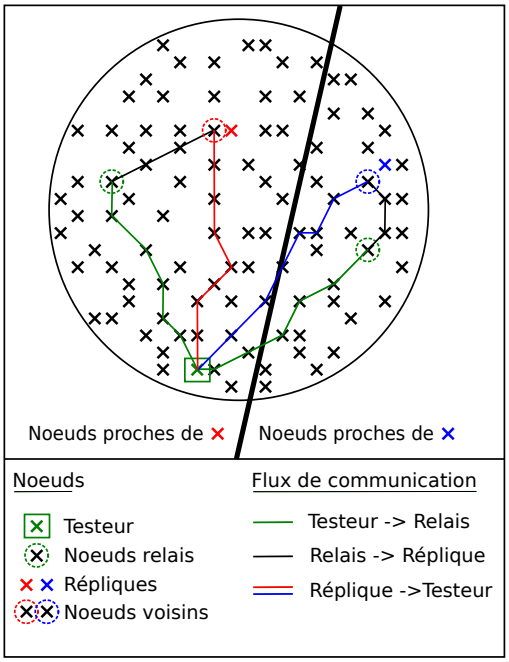

FIGURE 4.6 – Routage des requêtes de demande de position.

indépendamment des autres les étapes suivantes.

Protocole actif de découverte de répliques

1. Choisir au hasard k_1 nœuds à tester : $\alpha_1, \cdots, \alpha_{k_1} \in S$ et k_2 nœuds relais : $\beta_1, \cdots, \beta_{k_2} \in S$.
2. Pour i de 1 à k_1 créer une requête de demande de position géographique : `{LocRequest,`α_i`,`α`}`
3. Pour i de 1 à k_2 envoyer les requêtes de demande de position aux β_i
4. Attendre jusqu'à réception de k_2 déclarations de position à partir de chacun des nœuds testés ou bien jusqu'à ce que le temps d'attente soit épuisé.
5. Pour i de 1 à k_1 faire
   a. Si toutes les déclarations reçues des α_i sont cohérentes, alors ne rien faire
   b. Sinon choisir un sous-ensemble σ parmi les déclarations en conflit et diffuser dans tout le réseau la requête de révocation `{Revoke,` α_i`,` σ`}` afin de révoquer toutes les répliques de α_i

Trois cas de figure se présentent à chaque fois qu'un nœud γ reçoit une requête de demande de position {LocRequest, α_i, α}. Premièrement, si γ est le nœud α_i il établit une déclaration signée de sa position géographique {Position P, α_i}$_{K_{\alpha_i}^{-1}}$ et l'envoie à travers le voisin par lequel la requête a été reçue. Deuxièmement, si γ n'est ni α_i, ni un de ses voisins directs, il fait suivre la requête au nœud suivant de la route vers α_i. Troisièmement, si γ est un voisin direct de α_i, il suit le protocole suivant.

Protocole de récupération des déclarations de position

---

1. Demander à α_i de lui retourner sa déclaration de position signée
2. - Si α_i accepte et répond, alors il vérifie que la déclaration de position est valide
   - Si α_i refuse ou fournit une déclaration invalide, alors
   - Refuser de relayer les messages vers ou depuis α_i
   - Créer une requête pour isoler α_i, {Isolate, γ, {LocRequest, α_i, α}}
   - Diffuser localement cette requête d'isolation
   - Attendre une déclaration de position signée de α_i
   - Si une déclaration de position valide est obtenue, rétablir la communication avec α_i et continuer à exécuter le protocole
3. Envoyer la déclaration reçue vers α.

La déclaration de position du nœud α_i est dite valide si, d'une part, les coordonnées géographiques qu'elle transporte sont correctes, d'autre part, si sa signature est valide et authentique. Le nœud α_i répond à une requête de demande de position en diffusant localement sa déclaration signée. Les voisins sauvegardent les réponses reçues et ignoreront, durant le même tour d'exécution du protocole, les requêtes qu'ils recevront par la suite de α et ceci afin de réduire la quantité de trafic généré.

Quand un nœud γ' reçoit une requête d'isolation {Isolate, γ, {LocRequest, α_i, α}}, il vérifie tout d'abord s'il est un voisin de α_i. Si ce n'est pas le cas, la requête est ignorée. Sinon, γ' exécute à son tour le protocole de récupération des déclarations de position sauf qu'à la dernière étape du protocole, au lieu de retourner la déclaration signée à α, il l'envoie à γ.

La procédure d'isolation est récursive. En effet, si γ' n'est pas capable de récupérer la déclaration de position signée de α_i, il diffuse une nouvelle requête d'isolation afin de déléguer cette tâche à d'autres voisins. Si un de ces voisins arrive à obtenir la déclaration

signée de α_i, il la fait suivre à γ' qui à son tour la fait suivre à γ. La densité des réseaux de capteurs assure que la diffusion des requêtes d'isolation atteindront, après juste quelques itérations, tous les voisins de α_i. Si α_i refuse de collaborer, il sera de ce fait complètement isolé.

4.4.2 Analyse de sécurité

Dans cette section, nous allons prouver qu'un nœud répliqué a une probabilité exponentiellement petite en $min(k_1, k_2)$ d'éviter d'être détecté. Pour cela, nous allons commencer par prouver deux propriétés intermédiaires utiles à notre démonstration. Ces deux propriétés associent k_1 et k_2 à la probabilité pour un nœud quelconque du réseau d'être testé et à la probabilité pour un nœud répliqué, concerné par une vérification en cours, d'être découvert. Considérons le lemme suivant.

Lemme 3. *Quel que soit $k_1 > 0$, $U_n = (1 - k_1/n)^n$ est une suite croissante qui converge vers e^{-k_1}.*

Démonstration. Le Développement du terme général avec une formule binomiale démontre que $U_{n+1} > U_n$ et donc que la suite est croissante. Comme $(1 - k_1/n)^n = e^{n \ln(1-k_1/n)}$, en utilisant les séries de Taylor de $\ln(1 - x)$ on obtient $U_n = e^{n(-k_1/n+O((k_1/n)^2))} = e^{-k_1+O(k_1^2/n)}$ et par conséquent la limite de U_n est e^{-k_1}. □

Proposition 3. *La probabilité pour un nœud quelconque de ne pas être testé est exponentiellement petite en k_1.*

Démonstration. Un nœud choisit k_1 nœuds à tester parmi $n - 1$ (il ne se choisit pas lui-même). Comme les nœuds testés sont choisis uniformément, un nœud quelconque α aura une probabilité de $k_1/(n - 1)$ d'être testé par un autre nœud quelconque β du réseau. Comme chaque nœud opère indépendamment des autres, la probabilité qu'un nœud α ne soit testé par aucun autre nœud du réseau est $U_{n-1} = (1 - k_1/(n - 1))^{n-1}$. Comme U_n est une suite croissante, la probabilité pour un nœud quelconque de ne pas être testé est inférieure à e^{-k_1}, ce qui confirme donc notre proposition. □

Cette valeur décroît très rapidement même pour des valeurs pas très élevées de k_1. Par exemple, pour $k_1 = 3$ on aura une probabilité inférieure à 5% qu'un nœud quelconque ne soit pas concerné par un test et pour $k_1 = 5$, cette probabilité descend sous les 1%.

Lemme 4. *La probabilité pour que tous les relais délivrent la requête de demande de position à la même réplique est exponentiellement petite en k_2.*

Démonstration. Quand un nœud α est répliqué, on peut associer à chacune des répliques α_i l'ensemble des nœuds S_i les plus proches (du point de vue du routage) de α_i que des autres répliques. Les ensembles S_i forment une partition du réseau. Par exemple, si deux répliques du nœud α existent (α_1 et α_2), le réseau sera partitionné en deux : les nœuds qui vont router les messages au nœud α_1 et les nœuds qui vont router les messages vers α_2. Prenons n_p le nombre de nœuds dans la partition la plus grande. La probabilité qu'un nœud, choisi aléatoirement, soit dans cette partition est $n_p/n < 1$. Par conséquent, la probabilité pour que k_2 nœuds aléatoires soient dans la même partition est inférieure à $(n_p/n)^{k_2}$. La probabilité que toutes les requêtes de demande de position soient relayées vers la même réplique est donc $(n_p/n)^{k_2}$, ce qui est exponentiellement petit en k_2. □

Proposition 4. *Deux répliques d'un nœud, en cours de test, ont une probabilité exponentiellement petite en k_2 d'éviter d'être détectées.*

Démonstration. Supposons que α est un nœud répliqué en cours de test par un nœud β quelconque. Si β choisit au hasard k_2 nœuds relais qui vont router la requête de demande de position vers la même réplique, le protocole suivra son exécution mais sans détecter la réplique . Par ailleurs, si deux requêtes atteignent deux répliques différentes, β obtiendra la preuve de révocation ou bien l'une des deux répliques sera localement isolée par ses voisins. Pour que la détection échoue, tous les nœuds relais devront relayer leurs requêtes à la même réplique ce qui arrive d'après le lemme 4 très rarement c'est-à-dire avec une probabilité exponentiellement basse suivant k_2. □

La réplication d'un nœud sera ainsi détectée (globalement grâce à la preuve de révocation ou bien localement si une réplique refuse de coopérer) avec une probabilité exponentiellement proche de 1 en k_2. Si un nœud a une seule réplique et que chaque partition contienne $n/2$ nœuds, la probabilité qu'une réplique d'un nœud résiste à une détection est de $2 \times 1/2^{k_2}$ (toutes les requêtes dans une partition ou dans une autre). Pour $k_2 = 5$ cette probabilité est approximativement de 6% et descend sous la barre des 1% pour $k_2 = 7$.

Un nœud répliqué ne sera pas détecté s'il n'est testé par aucun nœud du réseau ou bien s'il est testé mais qu'il arrive à éviter la détection. Dans ces deux cas, la probabilité de non-détection est exponentiellement basse en k_1 ou bien en k_2. Dans notre protocole, la probabilité de non-détection d'un nœud répliqué est de ce fait exponentiellement basse en $min(k_1, k_2)$.

Chaque nœud choisit, à chaque round d'exécution, aléatoirement et indépendamment des autres nœuds, $k1$ nœuds à tester parmi les n nœuds du réseau, et ce, indépendamment de

leurs identités et de leurs positions géographiques. Le choix des relais se fait aussi de manière aléatoire. Un attaquant désirant neutraliser les témoins d'une réplique, n'aura, au moment de l'exécution de la détection, aucune information (qu'elle porte sur leurs identités ou sur leurs positions géographiques) ni sur les nœuds testeurs, ni sur les relais de ces nœuds testeurs. Par conséquent nous pouvons affirmer que notre protocole respecte le modèle d'attaquant que nous avons décrit précédemment (cf. section 4.2.4).

4.4.3 Analyse de performance

Résultats asymptotiques

Dans notre protocole, chaque requête génère trois types de transmission ; k_2 transmissions depuis le nœud testeur vers les nœuds relais ; $k_1 \times k_2$ transmissions depuis les nœuds relais vers les nœuds testés ; et k_1 transmissions depuis les nœuds testés vers le nœud testeur. La complexité en termes de communication est donc de $K\sqrt{n}$ messages par nœud avec $K = 1/2(k_2 + k_1k_2 + k_1)$.

Le protocole de récupération des déclarations de position et la procédure d'isolation génèrent ensemble au maximum $O(d)$ messages locaux, ce qui est négligeable en comparaison avec le protocole de détection de répliques.

Protocole	Communication	Mémoire
Diffusion nœud-réseau	$O(n)$	$O(d)$
Multicast déterministe	$O(\frac{g \ln g \times \sqrt{n}}{d})$	$O(g)$
Multicast stochastique	$O(n)$	$O(\sqrt{n})$
Line selected multicast	$O(\sqrt{n})$	$O(\sqrt{n})$
LMA	$O(\sqrt{n})$	$O(1)$
RED	$O(\sqrt{n})$	$O(1)$
Notre protocole	$O(\sqrt{n})$	$O(1)$

TABLE 4.1 – Analyse de performance asymptotique

En ce qui concerne la complexité en termes de calcul de notre protocole, un nœud a besoin de signer seulement une fois sa déclaration de position et peut l'utiliser à chaque fois

qu'elle est sollicitée par un nœud testeur. Par ailleurs, les signatures des déclarations sont vérifiées $2k_1k_2$ fois (une fois par chaque voisin à un saut du nœud testé et une fois par le nœud testeur lui-même et ceci pour chaque nœud testé).

Chaque nœud doit se souvenir des identités des nœuds qu'il teste, ce qui nécessite $O(k_1)$ de mémoire. Le nombre de déclarations signées traitées par nœud est de k_1k_2. Dans le cas le plus extrême, un nœud pourrait nécessiter $O(k_1k_2)$ déclarations signées en même temps, ce qui est peu probable mais possible.

Une comparaison des coûts en communication et en mémoire entre les protocoles de Parno et *al.*, les protocoles *RED* et *LMA*, et le notre est donnée dans le tableau 4.1. Par rapport au protocole *LSM*, les trois alternatives présentées dans ce chapitre permettent d'obtenir un gain significatif en mémoire tout en conservant les performances en communication. D'un point de vue asymptotique ces trois protocoles (*RED*, *LMA* et le notre) sont identiques au niveau des coûts. Pour trier leurs performances, il faut étudier les constantes multiplicatives, en particulier au niveau des communications. Les protocoles *RED* et *LMA* (pour sa version *SDC*) ne donnent lieu qu'à l'envoi d'une seule déclaration signée par nœud et donc à un nombre d'émissions par nœud très proche de $\sqrt{n}$. Notre protocole, quant à lui, a un coût plus important, de l'ordre de $k_1k_2\sqrt{n}$ pour des petites valeurs de k_1 et k_2. Ce facteur multiplicatif est le prix à payer pour les avantages au niveau de la sécurité et pour l'élimination d'une hypothèse comme celle de l'élection du leader. Nous évaluons la valeur de ce facteur multiplicatif dans ce qui suit.

Simulations

Afin d'avoir une idée sur le comportement de notre protocole dans un environnement réel et une estimation de ses performances dans la pratique, nous avons réalisé un ensemble de simulations en utilisant JiST, un simulateur utilisant la machine virtuelle Java [BHvR05]. Nous avons suivi la même procédure de simulation que Parno et *al.* et nous avons évalué le pourcentage de détection en utilisant les mêmes topologies réseaux qu'eux (voir Annexe B). Nous avons aussi évalué le nombre moyen de paquets envoyés et reçus par nœud, pour un réseau de n nœuds avec $n \in [1000; 10000]$. Les nœuds sont positionnés sur une zone géographique suivant une distribution uniforme avec une taille choisie de telle manière à ce que le nombre moyen de voisins d d'un nœud quelconque soit de 40.

On peut remarquer en analysant ces résultats de simulation, qu'en prenant $k_1 = 2$ et $k_2 = 3$, on atteint un taux moyen de détection d'environ 75% (voir Figure 4.7).

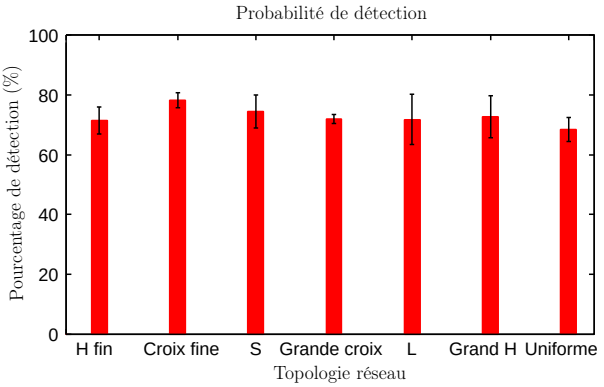

FIGURE 4.7 – Pourcentage de détection avec $k_1 = 2$ et $k_2 = 3$.

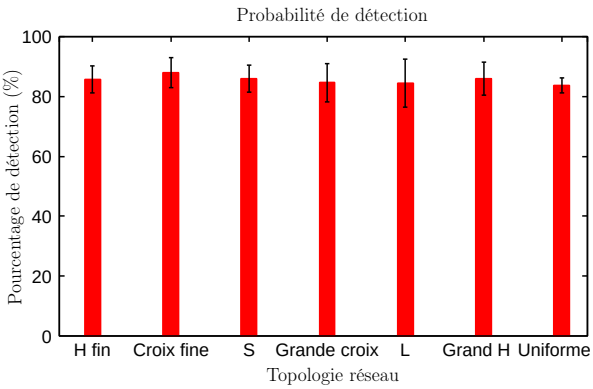

FIGURE 4.8 – Pourcentage de Détection avec $k_1 = 3$ et $k_2 = 3$.

L'augmentation du nombre de nœuds testés à $k_1 = 3$ permet d'augmenter le pourcentage de détection à environ 85% (voir Figure 4.8). Dans les deux jeux de tests, un seul nœud est répliqué avec seulement deux répliques. Dans le cas où le nombre de répliques augmenterait, le pourcentage de détection sera exponentiellement proche de cent pour cent (on peut en dire de même en ce qui concerne les autres protocoles).

Dans la figure 4.9, le coût en communication des différents protocoles est comparé. Le coût des protocoles *RED* et *LMA* (dans sa version SDC) est représenté par la courbe *SQRT(n)*. Le coût de *LMA* dans sa version P-MPC est proportionnel à celui de la version SDC en suivant une constante ayant un rapport avec la résistance aux attaques et non avec le taux de détection. Le choix de cette constante étant du point de vue de cette figure arbitraire, nous nous bornons à la représentation de *LMA* dans sa version SDC.

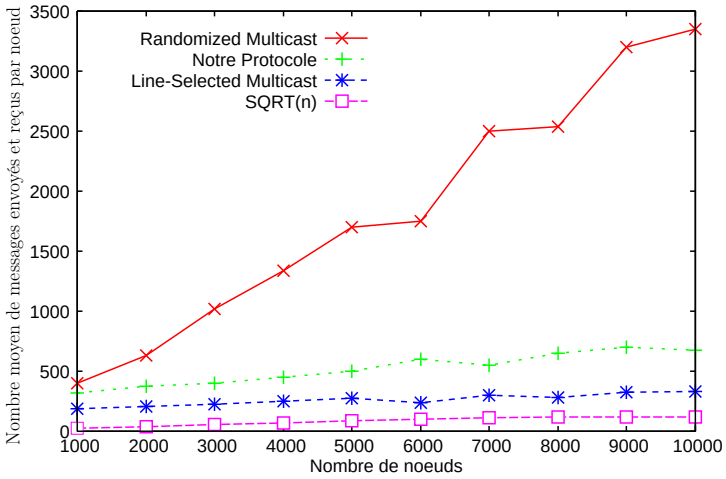

FIGURE 4.9 – Complexité en communication.

Le nombre moyen de paquets envoyés et reçus par nœud dans le protocole *LSM* de Parno et *al.* est de $3\sqrt{n}$. Dans notre protocole, en prenant les paramètres $k_1 = 2$ et $k_2 = 3$, le nombre moyen de paquets est $5.5\sqrt{n}$. La simulation confirme les résultats théoriques en ce qui concerne la complexité des communications, notre protocole possède une complexité de $O(\sqrt{n})$ avec un coût de communication légèrement supérieur à celui de Parno et *al.* et environ cinq fois plus important que ceux de *RED* et *LMA*.

4.5 Conclusion

Construire un protocole de détection d'une attaque par réplication de nœuds se fait sous certaines contraintes. Tout d'abord, les positions des nœuds doivent être portées à la connaissance d'un certain nombre de nœuds témoins. Deuxièmement, il faut au moins qu'un des nœuds témoins obtienne deux déclarations de position contradictoires, dans le cas où un nœud est réellement répliqué dans le réseau. Troisièmement, les nœuds témoins doivent être aléatoirement associés aux nœuds testés afin que le protocole soit résistant à un maximum de types d'attaques. Enfin, il faut prendre le minimum de nœuds témoins possibles et les atteindre en générant un minimum de communications afin de tenir compte des considérations de performance du protocole.

D'une certaine façon, il est possible de voir une exécution de notre protocole comme la création d'une série de *wormholes* (en référence à l'attaque décrite en section 2.3.2.1) entre le

testeur et ses relais. Ces wormholes permettent au testeur d'avoir un œil à différents endroits du réseau et donc d'être quelque part un témoin multicéphale.

Notre protocole permet de garantir avec une forte probabilité que le nœud testeur aura deux déclarations contradictoires si des répliques existent. Les relais utilisés par le nœud testeur, ainsi que les nœuds testés étant choisis de façon uniformément aléatoire, la résistance aux attaques est maximale. Enfin, le nombre de messages pour atteindre les témoins est asymptotiquement optimal et en pratique à un petit facteur près (inférieur à dix) de l'optimalité.

Ce protocole permet d'atteindre ces performances sans faire des hypothèses contraignantes comme c'est le cas pour le protocole RED, au prix d'un coût légèrement supérieur par rapport au nombre de messages émis pour atteindre les témoins. Il nous semble difficile d'envisager un protocole permettant d'atteindre les performances du multicast déterministe avec un seul témoin tout en étant sûr, autrement qu'en utilisant une approche comme celle de RED : choisir la méthode de sélection des témoins à la dernière minute. Réciproquement, il nous semble difficile d'envisager une méthode collaborative de choix avec des coûts raisonnables et résistant aux attaques.

Sans atteindre *précisément* le coût du multicast déterministe à un témoin, il est peut-être possible de se rapprocher de ces coûts d'un facteur inférieur à cinq tout en ayant un taux de détection élevé. D'autres protocoles de détection actifs sont une approche qui nous semble intéressante pour atteindre ce but.

Par ailleurs, le protocole de détection actif que nous avons choisi ici est basé sur l'hypothèse que les routes entre les nœuds existent grâce à l'utilisation d'un protocole de routage ad hoc, ce qui engendre des frais de routage supplémentaires que nous n'avons pas évalués dans ce manuscrit. À cet effet, un de nos projets futurs est d'améliorer notre protocole de détection actif en axant aussi nos efforts sur le processus d'établissement de routes.

Chapitre 5

Conclusion et Perspectives

Nous pouvons distinguer deux approches de sécurisation des réseaux ad hoc. Une approche reposant sur des techniques cryptographiques. Cette approche est souvent utilisée comme un moyen de prévention et constitue souvent le premier rempart de sécurité du réseau, mais peut s'avérer parfois insuffisante contre certains comportements malveillants des nœuds. Une deuxième approche très intéressante contre ce type d'adversaire consiste à utiliser les systèmes de confiance et de réputation. Cependant, cette deuxième approche est sujette à certaines menaces. Nous nous sommes intéressés ici à deux d'entre elles : la protection des données privées et la gestion des identités.

Nous avons apporté une solution au problème de protection de la vie privée dans les systèmes de gestion de confiance en choisissant de protéger les données privatives des différents participants au système plutôt que de s'appuyer sur l'anonymisation des identités, et ce, pour les raisons que nous avons citées dans le chapitre 3.

Après avoir présenté les travaux proposés dans le cadre de la préservation des données privatives dans les systèmes de confiance, nous avons proposé une solution efficace, dont la sécurité a été prouvée dans le modèle semi-honnête et qui est optimale en termes de résistance aux collusions. Le schéma que nous avons proposé est beaucoup plus sûr comparé à celui proposé par Yao et *al.* (la solution que nous avons prise comme référence à nos comparaisons). Effectivement, notre protocole permet une complète résistance aux collusions dans le sens où il suffit qu'un seul participant soit honnête pour que le protocole soit sûr dans le cadre du modèle semi-honnête. S'agissant de la sécurité de notre protocole dans un modèle d'attaquant malveillant, notre protocole peut facilement être généralisé pour qu'il puisse prendre en compte de nombreux comportements malveillants. Il est aussi plus efficace notamment en termes de coûts de communication que ses prédécesseurs.

Il reste, néanmoins, certaines améliorations à apporter au protocole et que nous espérons concrétiser dans nos futurs travaux. Il est question, dans un premier temps, d'étendre notre protocole afin qu'il tienne compte de la majorité des comportements malveillants et, dans un second temps, d'implémenter notre protocole dans un vrai système de confiance.

Nous avons aussi apporté une solution au problème de la réplication des identités dans le contexte des réseaux de capteurs. Sachant que les capteurs sont extrêmement limités en ressources matérielles, il est donc important de tenir compte de ce fait dans le développement de solutions logicielles destinées à ce type de réseaux et ceci est évidemment valable dans notre situation.

L'attaque par réplication de nœuds est un cas particulier de l'attaque Sybille qui apparaît dans certains réseaux à faible protection physique tels que les réseaux de capteurs. Un attaquant peut capturer un nœud, le dupliquer autant de fois qu'il le désire et utiliser ces répliques afin d'avoir un accès au réseau et mener d'autres attaques plus nuisibles ou plus profitables.

Nous avons présenté dans ce manuscrit une solution au problème de la réplication des nœuds. Nous avons réussi à développer une solution robuste donnant des résultats en termes de détection et de performance très satisfaisants que ce soit dans le cas d'évaluations théoriques ou au cours des différentes simulations réalisées.

Notre protocole utilise une approche de détection différente de celles de Parno et al, du protocole RED (ce protocole étant le plus résistant aux attaques) et LMA, ce qui lui permet d'avoir les mêmes avantages en termes de résistance aux attaques que le protocole RED, mais sans avoir ses inconvénients (c'est-à-dire, il ne fait appel ni aux services d'une tierce partie de confiance, ni à un protocole additionnel moins sûr). De plus, il offre les mêmes performances que les protocoles RED et LMA en termes de communication et d'utilisation mémoire.

Bien sûr, d'autres variantes de notre approche sont envisageables et notre objectif à court terme est de trouver d'autres solutions qui minimisent davantage les coûts en termes de communication et de calcul.

Annexes

Annexe A

Protocole de routage hybride basé sur les colonies de fourmis

Nous avons présenté dans le chapitre 2 plusieurs architectures de réseau ad hoc en faisant ressortir leurs caractéristiques communes à savoir le manque de ressources matérielles, la mobilité et le caractère autonome et spontané de ce type de réseaux. Ces caractéristiques engendrent de nouvelles problématiques qui n'existaient pas dans les réseaux filaires traditionnels. Une des problématiques est le routage des données.

Nous pouvons généralement distinguer deux catégories de protocoles de routage : les protocoles utilisant une approche proactive et les protocoles utilisant une approche réactive. Que l'on utilise une approche réactive ou proactive pour un protocole de routage, deux problèmes sont récurrents.

Le premier problème que nous allons soulever est celui du surcoût de communication causé par l'activité des protocoles de routage utilisés habituellement. Ce surcoût est dû essentiellement soit à la technique de broadcast utilisée dans le cas des protocoles réactifs (AODV, ARA, etc.), soit aux échanges périodiques des tables de routage, de taille non négligeable, dans le cas des protocoles de routage purement proactifs (DSDV, OLSR, etc.).

Le deuxième problème est qu'indépendamment du coût de communication, l'introduction de nombreux paquets de contrôle peut entraîner une congestion du réseau. En effet, lorsqu'il y a trop de paquets présents sur le réseau, ou tout du moins sur une partie de celui-ci, les performances se dégradent ; il se produit une congestion. Lorsque le nombre de paquets émis par les nœuds reste dans les limites de la capacité de transport du

réseau, tous les paquets seront correctement délivrés à leurs destinataires, à l'exception de quelques-uns ayant souffert d'une erreur de transmission. Toutefois, si le trafic augmente fortement, les nœuds ne sont plus capables de faire face à la demande et commencent à perdre des paquets. Cela a tendance à aggraver la situation. Lorsque le trafic atteint des crêtes, les performances chutent littéralement et très peu de paquets sont délivrés. De plus, comme les nœuds d'un réseau ad hoc ont des ressources matérielles limitées, cela accélère davantage le phénomène de congestion.

Il existe différents facteurs possibles de congestion. Si la mémoire des nœuds n'est pas suffisante, des paquets seront perdus. L'ajout de mémoire peut avoir un effet positif, mais jusqu'à un certain point seulement. En effet, lorsque les nœuds ont une quantité de mémoire infinie, cela peut provoquer un effet inverse et la situation empire. Effectivement, avant que les paquets ne parviennent en tête de file d'attente, ils ont déjà eu le temps d'expirer et même plusieurs fois, ce qui augmente encore davantage la congestion. Les processeurs lents peuvent aussi être une source de congestion. Si des nœuds souffrent de la lenteur de leur CPU et exécutent lentement les tâches de gestion du trafic entrant et sortant, des files d'attente se forment. De même, les liens à faible bande passante peuvent aussi provoquer un encombrement.

Afin d'apporter une solution partielle à ces problèmes, nous proposons un protocole hybride dans le sens où il combine une construction périodique de routes et une construction de routes à la demande. En effet, il ne s'agit plus dans notre protocole d'échange d'informations volumineuses telles que les tables de routage, mais d'agents "utiles" qui contiennent le minimum d'informations possibles. Notre protocole de routage est multiagent, chaque nœud met périodiquement un certain nombre d'agents (fourmis) à la disposition du réseau. Chaque agent permet, d'une part, une construction stochastique de routes vers le nœud qui l'a généré (état proactif de l'agent) et d'autre part, d'aider les autres nœuds dans leurs processus respectifs d'établissement de routes (état réactif de l'agent).

Grâce à l'état proactif des agents, un pourcentage important de routes sont construites. L'idée est d'utiliser cette quantité d'informations afin de construire, au besoin, les routes restantes. Une première idée est d'utiliser la technique de diffusion comme c'est le cas dans les protocoles réactifs. Mais vu le nombre important de routes construites durant la phase proactive, l'exploration totale de tout le réseau est, dans certaines situations (topologie, densité, etc.), une solution qui peut s'avérer non nécessaire. Notre idée est de faire une exploration plus intelligente du réseau en s'appuyant sur le principe de fonctionnement des fourmis. En effet, au lieu que la demande de route soit diffusée dans tout le réseau, comme

c'est le cas notamment dans les protocoles de routage existants basés sur les colonies de fourmis [Bou02, GFLM05], elle se fera localement au niveau de chaque nœud et c'est aux agents de la disséminer. À cet effet, nous proposons un nouveau schéma de découverte de routes moins coûteux, qui s'adapte au caractère dynamique des réseaux ad hoc et à l'activité du réseau tout en étant plus efficace dans la pratique que le broadcast.

Le choix d'utiliser une approche inspirée du fonctionnement des colonies de fourmis est motivé par le fait que ces protocoles s'adaptent bien aux changements de topologie et dans la majorité des cas génèrent des coûts en termes de communication moins importants en comparaison avec les protocoles de routage traditionnels (proactifs et réactifs) (voir quelques comparaisons de performance dans [Bou02, GFLM05]).

Notre protocole ne permet pas d'établir systématiquement les routes les plus courtes. Par contre, il permet à un nœud de posséder plusieurs routes vers une même destination. Ceci a plusieurs avantages comme une meilleure robustesse face aux changements de topologie et la possibilité de choisir une route optimale par rapport à différents paramètres comme la qualité des liens ou le degré de fiabilité des nœuds intermédiaires.

Afin de valider notre schéma et d'évaluer ses performances, nous avons opté pour un réseau de densité et de taille moyenne. Notre choix est motivé par le fait que ce type de réseau est le plus répandu dans la réalité. Nous l'avons comparé à deux protocoles : un protocole réactif (AODV) et un protocole proactif (DSDV). Les résultats obtenus en utilisant une première version de notre protocole, confirment nos prévisions. En effet, nos résultats sont semblables aux résultats du protocole DSDV qui est réputé pour être très à l'aise dans ce genre de réseaux.

Il faut aussi noter que cette première version est obtenue après plusieurs essais en faisant varier certains paramètres importants du protocole : la quantité initiale de phéromone, la quantité de phéromone à décrémenter lors du processus d'évaporation, la fréquence initiale d'envoi des agents, la valeur du TTL, etc.

A.1 Protocoles de routage basés sur le fonctionnement des colonies de fourmis

Cette catégorie de protocoles de routage s'est inspirée du fonctionnement des colonies de fourmis. Elle s'appuie sur des algorithmes fondés sur la capacité de simples fourmis (agents) à résoudre des problèmes complexes par coopération. Les algorithmes basés sur les colonies

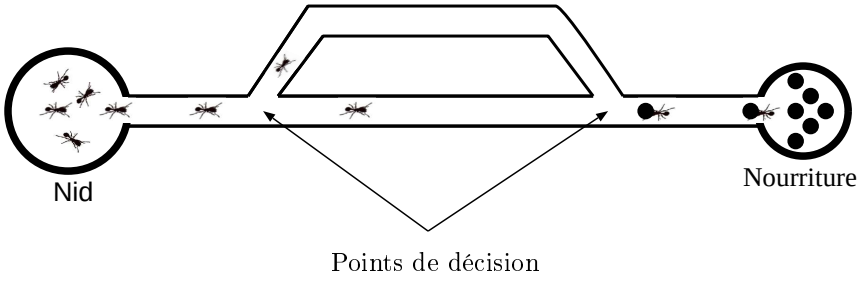

FIGURE A.1 – Fonctionnement des colonies de fourmis dans la recherche de nourriture

de fourmis constituent un sous-ensemble de ce qu'on appelle l'*intelligence en essaim*[1] [BDT99]. L'idée principale de ces protocoles est de construire des chemins (routes) entre des nœuds sources et destinations en imitant les fourmis dans leur tâche de recherche de nourriture. Un point très important de cette méthode est que les agents (fourmis) n'ont pas besoin de communiquer directement pour accomplir leur tâche, mais ils le font indirectement à travers la modification de leur environnement par dépôt de phéromones[2].

Il a été démontré, à travers le développement de protocoles de routage basés sur ce principe (p. ex. [Bou02, GFLM05]), que ce type d'algorithmes s'adapte très bien au caractère autonome des réseaux ad hoc. De plus, ces algorithmes ont la particularité de réagir rapidement aux changements fréquents dans la topologie des réseaux ad hoc, ce qui facilite par conséquent la maintenance des routes. Un autre avantage de ces protocoles est qu'ils génèrent peu de trafic réseau (paquets de contrôle) comparés aux protocoles traditionnels tels que AODV ou OLSR.

A.1.1 Algorithme de fourmis de base

L'idée de base des protocoles de routage fondés sur les colonies de fourmis est inspirée du comportement de fourmis réelles lorsqu'elles recherchent de la nourriture. Quand les fourmis sont à la cherche de nourriture, elles se hasardent dans la nature et au cours de leurs déplacements, elles déposent des quantités constantes de phéromones. Quand elles rencontrent une intersection (points de décision sur la figure A.1) formée par un obstacle, elles décident de la route à suivre. Ce choix se fait de façon stochastique et

[1] L'intelligence en essaim est une propriété de systèmes de robots non intelligents qui montrent collectivement un comportement intelligent.

[2] Substance chimique qui, émise à dose infime par un animal (p. ex. les fourmis) dans le milieu extérieur, provoque chez ses congénères des comportements spécifiques.

proportionnellement à la quantité de phéromones présente sur chaque route. Avec le temps, la concentration des phéromones diminue, ceci est dû à un phénomène d'évaporation appelé diffusion. Ce phénomène est très utile aux fourmis et leur permet de tenir compte des différents changements qui s'opèrent dans leur environnement (nouveaux obstacles par exemple). C'est justement ce processus qui est repris tel quel dans les protocoles de routage basés sur les colonies de fourmis pour gérer les changements de topologie dans les réseaux ad hoc.

Afin de mieux comprendre la dynamique du processus de recherche de nourriture chez les fourmis, nous allons prendre un exemple faisant état de deux routes entre le nid et la nourriture Figure A.1. Arrivée à la première intersection, la première fourmi choisit aléatoirement une branche. Comme la route de dessous est la plus courte, les fourmis qui auront choisi cette route atteindront la nourriture les premières. Et à leur retour vers le nid, elles vont choisir aussi cette même route. Après un certain temps, la concentration de phéromones dans le chemin le plus court sera plus importante. Une fois le plus court chemin identifié, toutes les fourmis l'emprunteront.

A.1.2 ACO : Ant Colony Optimization meta-heuristic

Le point commun entre les différents protocoles de routage ad hoc basés sur le fonctionnement des fourmis est qu'ils utilisent une adaptation de l'heuristique ACO [DDC99] détaillée ci-dessous.

Soit $G = (V, E)$ un graphe connexe avec $n = |V|$ nœuds.

ACO est utilisée pour trouver le plus court chemin entre une source v_s et une destination v_d d'un graphe G. On trouve sur chaque arc de G une quantité de phéromone ($\forall\ e(i,j) \in E$ *reliant* v_i *et* v_j, *on a* $e(i,j) \leftarrow \varphi_{i,j}$) modifiée à chaque passage d'une fourmi ; la quantité de phéromone nous renseigne sur le nombre de fourmis qui ont emprunté cet arc.

Une fourmi se trouvant sur le nœud v_i utilise la quantité de phéromone $\varphi_{i,j}$ associée au nœud $v_j \in N_i$ pour calculer la probabilité de le choisir comme nœud suivant. N_i est l'ensemble des voisins à un saut de v_i :

$$p_{i,j} = \frac{\varphi_{i,j}}{\sum_{v_j \in N_i} \varphi_{i,j}} \qquad si\ v_j \in N_i;\ 0\ sinon, \quad \sum p_{i,j} = 1.$$

La fourmi modifie la quantité de phéromone d'un arc $e(i,j)$ lorsqu'elle quitte le nœud v_i pour aller vers le nœud v_j comme suit :

$$\varphi_{i,j} \leftarrow \varphi_{i,j} + \Delta\varphi.$$

Comme dans le cas des fourmis réelles, la concentration de phéromones diminue avec le temps. Dans l'heuristique ACO, elle se fait de manière exponentielle :

$$\varphi_{i,j} \leftarrow (1-q)\varphi_{i,j}, \qquad q \in [0,1].$$

Malgré qu'il est difficile de mener des analyses théoriques précises sur les protocoles de routage reposant sur l'heuristique ACO, à cause des nombreux paramètres aléatoires qui dépendent les uns des autres et dont la distribution change à chaque itération, ces protocoles donnent des résultats intéressants dans la pratique et en particulier en termes de réduction des coûts de communication et de réactivité aux changements de topologie. Elle représente donc une approche très intéressante à étudier.

A.1.3 ARA : The Ant-Colony Based Routing Algorithm for MANETs

Il existe de nombreux protocoles de routage utilisant l'heuristique ACO. On peut citer par exemple les travaux de Di Caro et *al.* [GFLM05] et le protocole ARA (The Ant-Colony Based Routing Algorithm) [Bou02]. Afin de comprendre la manière avec laquelle l'heuristique ACO est utilisée dans le cadre des protocoles de routage dans les réseaux ad hoc, nous prenons le protocole ARA comme exemple. On distingue trois phases principales :

Découverte de routes Durant cette phase, une nouvelle route est créée. La création d'une nouvelle route requiert l'utilisation de deux agents : un *forward ant* (FANT) et un *backward ant* (BANT). Un FANT est un petit paquet avec un unique numéro de séquence, ce qui va permettre aux nœuds de détecter les éventuelles copies d'un même FANT.

Quand un nœud source veut commencer une communication avec un nœud destination, et qu'il n'y a pas de route vers cette destination dans sa table de routage, il diffuse un FANT vers tous ses voisins à un saut. Un nœud qui va recevoir pour la première fois un FANT, crée une entrée dans sa table de routage. Une entrée dans la table de routage est un triplet (adresse destination, nœud suivant, quantité de phéromones). Le nœud interprète l'adresse de la source comme étant une destination, le nœud précédent comme étant le nœud suivant, et calcule la quantité de phéromones en prenant en considération le nombre de sauts entre la source et la destination. Ensuite, le nœud diffuse à son tour le FANT.

Quand le FANT atteint la destination, la destination crée un BANT et l'envoie vers le nœud source. Le BANT procède de la même façon que le FANT, c'est-à-dire il établit une route vers le nœud destination. Quand le nœud source reçoit le BANT, il aura la route vers

la destination, il ne lui restera qu'à envoyer le paquet de données vers la destination.

Maintenance de la route ARA n'a pas besoin de paquets spéciaux afin de maintenir les routes. Une fois le FANT et le BANT ont établi les routes vers les nœuds sources et destinations, les paquets de données sont utilisés pour la maintenance de la route. Quand un nœud v_i relaye un paquet de données vers $v_{destination}$ à travers v_j, il augmente la quantité de phéromones de l'entrée $(v_{destination}, v_j, \varphi)$ par $\Delta\varphi$, et le nœud suivant v_j, quant à lui, augmente la quantité de phéromone de l'entrée $(v_{source}, v_i, \varphi)$ par $\Delta\varphi$.

Le processus d'évaporation est simulé par une diminution régulière de la quantité de phéromone de chaque entrée.

Traitement des ruptures de liens La dernière phase est le traitement des ruptures de liens détectées en l'absence d'acquittement, lors de l'envoi d'un paquet de données. Si un nœud reçoit un message de type ROUTE_ERROR via un certain lien, il désactive en premier lieu ce lien en mettant la quantité de phéromone à 0. Ensuite, il recherche dans sa table de routage un autre lien lui permettant d'atteindre cette destination. Si ce lien existe, alors il envoie le paquet suivant cette route, sinon il informe ses voisins s'ils peuvent relayer ce paquet. Le processus est réitéré par chaque voisin. Le paquet peut, soit atteindre la destination ou bien, dans le cas contraire, la source renouvelle sa demande de découverte de route.

A.2 Notre proposition : Description

Avant de donner le détail sur le fonctionnement de notre protocole de routage, considérons quelques hypothèses de travail. Chaque nœud doit être capable de diffuser des messages *hello* à ses voisins à un saut et les liens entre les nœuds du réseau doivent toujours être bi-directionnels.

Notre protocole, que nous avons appelé *AntTrust*, est avant tout un protocole de routage. Le but d'un protocole de routage est de construire des routes optimales en termes de distance par exemple, entre des nœuds sources et destinations, tout en étant fortement réactif aux changements de topologie. Notre protocole de routage sera composé de trois modules :

1. module de découverte du voisinage ;
2. module de découverte de routes ;
3. module de gestion des ruptures de liens.

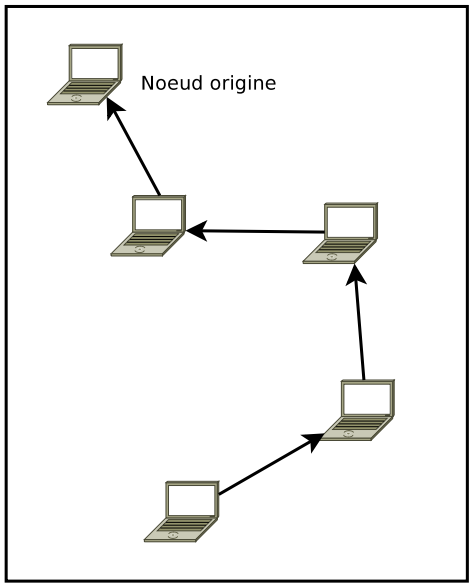

FIGURE A.2 – Phase aller du protocole AntTrust

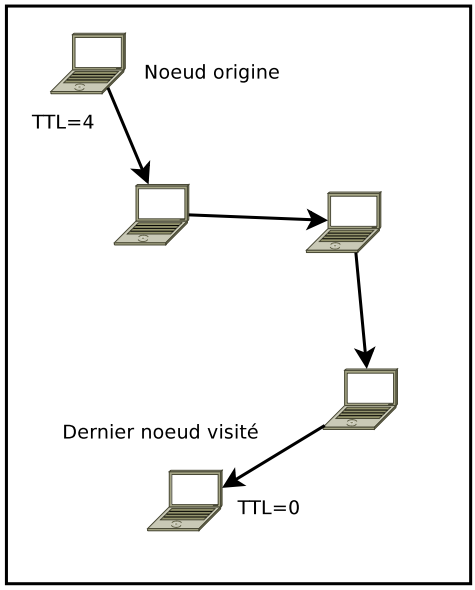

FIGURE A.3 – Phase retour du protocole AntTrsut

Nous allons omettre de donner ici la description du premier module qui est le même que dans les autres protocoles de routage. Le but d'un tel processus est de permettre à chaque nœud de connaître l'identité de ses voisins à un saut.

A.2.1 Découverte des routes

A.2.1.1 Phase proactive

Afin d'établir des routes entre les nœuds du réseau, notre protocole s'appuie sur des agents mobiles que nous avons appelés *Ant*. Chaque nœud du réseau génère périodiquement un agent *Ant* qui passe par deux phases distinctes durant son cycle de vie : une phase aller (figure A.2) et une phase retour (figure A.3). Chaque agent appartient à un nœud unique appelé *nœud origine*. Il faut aussi noter que tous les agents *Ant* transportent un numéro de séquence unique mis à jour fréquemment (p. ex., à chaque envoi d'un nouvel agent) par le nœud propriétaire de l'agent (*nœud origine*). Ce numéro de séquence permet aux nœuds de sélectionner les informations les plus récentes lorsqu'ils mettent à jour leurs tables de routage.

Les agents *Ant* se déplacent aléatoirement d'un nœud à un autre tout en construisant un

chemin vers leurs *nœuds origines* (voir figure A.2). Afin d'éviter la formation de boucles de routage, nous associons un identifiant unique à chaque agent créé par un nœud. Si un nœud reçoit plusieurs fois le même agent, il accepte les informations de routage apportées lors du premier passage de l'agent et ignore toutes les autres.

Afin de gérer leurs agents et en même temps construire des routes qui s'étendent sur toute la largeur du réseau, chaque nœud attribue un temps d'expiration (Time To Live (TTL)) aux agents créés. La valeur d'un TTL est déterminée suivant la dimension du réseau de telle sorte à ce que l'agent d'un nœud se trouvant dans une des extrémités du réseau puisse atteindre un nœud se trouvant à l'autre extrémité.

Durant sa phase retour, un agent suit le chemin inverse à celui qui l'a emprunté durant sa phase aller, et ce, jusqu'à son *nœud origine*. Durant cette phase retour, il construit des routes à partir des nœuds qu'il traverse vers le dernier nœud visité durant la première phase (p. ex. quand le $TTL = 0$).

Un résumé de cette première étape est donné dans l'algorithme suivant :

Phase proactive

Soient :

- N l'ensemble des nœuds du réseau
- $V(x) = \{x \, voisin \, de \, y\}$ l'ensemble des voisins du nœud x
- $Route(x, y)$ une route du nœud x vers le nœud y
- $RouteOptimale(x, y)$ la meilleure route d'un nœud x vers un y

Périodiquement :

Pour $i = 1, ..., N$ **faire**

1. le nœud n_0^i crée un agent Ant_i
2. **pour** $j = 1, ..., TTL$ **faire**
   - Ant_i choisit le nœud suivant n_j^i avec une probabilité uniforme :
     - $p(n_j^i) = \frac{1}{|V(n_{j-1}^i)|}$
   - Ant_i est porteur de la meilleure route possible vers n_0^i : $RouteOptimale(n_{j-1}^i, n_0^i)$

3. Ant_i bascule vers sa *phase retour* en suivant le chemin inverse de sa *phase aller* :
   il construit l'ensemble des routes suivant :
   $\{Route(n^i_{TTL-1}, n^i_{TTL}), Route(n^i_{TTL-2}, n^i_{TTL}), ..., Route(n^i_0, n^i_{TTL})\}$

Remarque 2. *Nous pouvons remarquer que les routes découvertes ne sont pas nécessairement les meilleures qui puissent exister dans le réseau. Cependant, il en existe plusieurs pour une même destination. Cette caractéristique est très appréciable en pratique, comme nous allons le constater dans les résultats de simulation. Elle permet, d'une part, aux utilisateurs de choisir entre différentes routes selon certains critères comme la confiance accordée aux voisins, d'autre part, de remédier aux fréquents changements dans la topologie du réseau. On peut aussi remarquer qu'en augmentant le nombre d'agents dans le réseau, on augmente en même temps le panel de routes dont un nœud pourrait disposer.*

A.2.1.2 Phase réactive

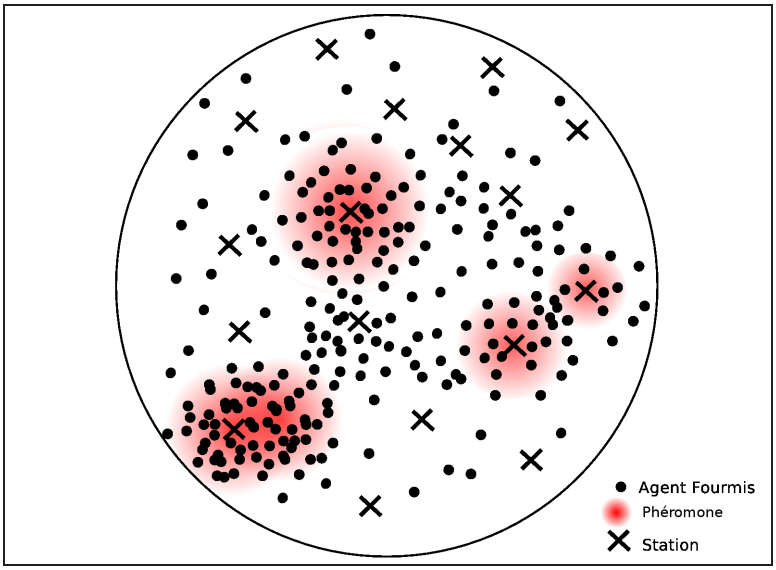

FIGURE A.4 – Influence des phéromones sur le mouvement des agents. Nous pouvons remarquer sur cette figure, que plus il y a de phéromones dans une zone donnée, plus il y a d'agents qui se dirigent vers cette zone.

Durant le premier processus proactif, un pourcentage considérable de routes ont été construites. Mais, quand un nœud veut envoyer un paquet de données à une destination donnée et qu'il ne dispose pas encore d'un chemin vers cette destination, il initie une demande de route explicite vers celle-ci. La première idée que nous avons envisagée est d'utiliser la technique de diffusion traditionnelle qui consiste à explorer tout le réseau dans le but de trouver une route vers cette destination. Cette première approche souffre d'un surcoût de communication et tout particulièrement dans le cas des réseaux denses. Comme il existe un certain nombre de routes préalablement établies, nous allons exploiter ces informations de routage afin de faire une exploration plus intelligente du réseau.

La demande de route n'est plus diffusée dans tout le réseau à l'instar de ce qui est fait dans les protocoles de routage traditionnels, mais elle est tout simplement déposée au niveau du nœud demandeur (le nœud tient à jour une table de demande de routes) et la tâche de diffusion revient aux agents circulant dans le réseau qui vont répandre cette information de façon "intelligente". Effectivement, durant leurs phases allers et dès qu'ils traversent un nœud qui fait une demande de route, les agents basculent vers leurs phases retours. Durant cette phase retour, les agents déposent des quantités de phéromones (voir section A.1) sur les nœuds qu'ils traversent jusqu'à leurs *nœuds origines*. Ce dépôt de phéromones va contribuer à marquer les routes vers les nœuds demandeurs, ce qui permettra d'attirer plus d'agents qui vont de manière coopérative aider à l'établissement des routes demandées.

Les agents se déplacent dans le réseau de manière stochastique et proportionnellement à la quantité de phéromone se trouvant sur chaque nœud (voir figure A.4). Autrement dit, un agent choisit le nœud suivant à visiter parmi les voisins du nœud courant de façon aléatoire et en donnant la priorité aux voisins menant vers des nœuds demandeurs de routes. Ce processus augmente les chances de choisir une direction vers un nœud demandeur tout en laissant des chances aux autres directions.

L'expression qui donne la probabilité de choisir comme nœud suivant le nœud n_j menant vers un nœud demandeur de route d alors que l'agent se trouve sur le nœud n_i est la suivante :

Soient $V(n_i) = \{n_j \textit{ voisin de } n_i\}$ et $Q_{(n_k,d_m)}$ la quantité de phéromone associée au voisin n_k de n_i pour la demande de route d_m :

$$p((n_j,d)) = Q_{(n_j,d)} / \sum_{n_k \in V(n_i)} Q_{(n_k,d_m)}. \tag{A.1}$$

Si le nœud suivant choisi mène vers un demandeur de route, l'agent vérifie si le nœud courant possède une route vers la destination recherchée par le demandeur. Si cette

route existe, il l'étend jusqu'au nœud demandeur. Dans le cas contraire, il continue son déplacement en se dirigeant cette fois-ci de manière déterministe vers le nœud demandeur, tout en essayant de trouver une route vers la destination recherchée, et ce, en consultant les tables de routage des nœuds intermédiaires. Une fois arrivé au nœud demandeur, l'agent exécute les instructions suivantes :

- s'il est porteur d'une route alors, il retourne à son nœud créateur (phase retour) ;
- sinon il vérifie si la demande est toujours d'actualité. Si c'est le cas, il retourne à son nœud d'origine tout en incrémentant les quantités de phéromone associées au demandeur. Dans le cas contraire, il retourne à son nœud créateur sans incrémenter les quantités de phéromone.

Afin de permettre aux demandes d'être prises en compte de manière équitable, et une meilleure gestion des changements de topologie, un processus d'évaporation est mis en place. À chaque intervalle de temps, la quantité de phéromone correspondant à chaque demande de route (chaque entrée de la table de demande de route, au niveau de chaque nœud) est décrémentée selon une certaine quantité.

Comme nous venons de le présenter, le déplacement des agents dépend de la quantité de phéromone au niveau de chaque nœud. Cette quantité de phéromone représente par ailleurs le nombre de demandes de route, ce qui implique une distribution intelligente des agents dans le réseau suivant les besoins de chaque zone. Cette propriété est très intéressante, car elle permet au protocole de s'adapter à l'activité des nœuds et de ce fait d'exploiter au mieux les capacités des agents.

Les phéromones sont aussi utilisées dans le but de gérer le nombre d'agents dans le réseau et donc de maîtriser la congestion du réseau.

À cet effet et grâce à l'utilisation des phéromones, notre protocole dispose implicitement d'un moyen pour lutter contre la congestion due aux paquets de contrôle. Pour ce faire et afin de trouver un équilibre optimal entre la fréquence d'envoi des agents et l'activité réseau, notre protocole mesure le rapport entre la quantité de phéromone et le nombre d'agents dans le réseau. Notre idée est d'exploiter le déplacement des agents afin de récolter des informations concernant le nombre d'agents dans le réseau ainsi que la quantité de phéromone au niveau de chaque nœud. Cette mesure permet à chaque nœud d'estimer localement les besoins du réseau en termes de communication et d'ajuster en conséquence

sa fréquence d'envoi d'agents.

Un autre rôle important des agents est celui de la dissémination des routes optimales, ce qui signifie en d'autres termes, choisir les entrées des tables de routage les plus intéressantes. Une entrée d'une table de routage est de la forme :

<Destination, Noeud suivant, Distance (métrique d'évaluation de la meilleure route), Liste des voisins concernés par la route, Numéro de séquence>.

Quand un agent arrive sur un nœud, il met à jour les informations de routage locales et il rafraîchit en même temps ses propres informations de routage en puisant dans les informations se trouvant sur ce nœud. La sélection d'une route optimale se fait, tout d'abord, sur la base du plus grand *numéro de séquence* qui correspond aux routes les plus récentes et ensuite sur la base de la plus courte distance en termes de nombre de sauts.

A.2.2 Gestion des ruptures de liens

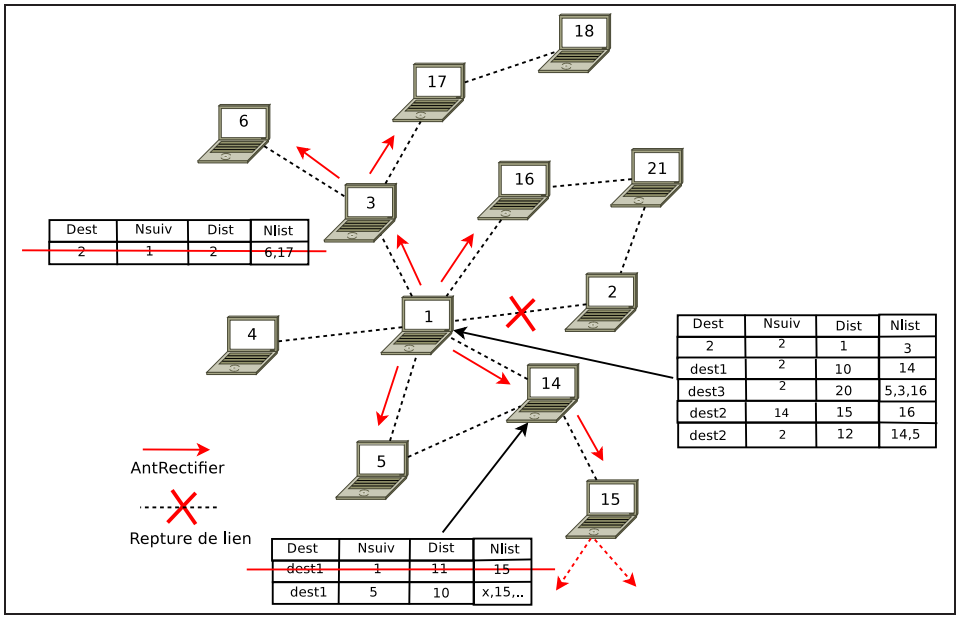

FIGURE A.5 – Gestion des ruptures de liens

Les ruptures de liens sont gérées au niveau de chaque nœud par un agent spécial

que nous avons dénommé *RectifierAnt*. Ces agents sont créés par un nœud à chaque fois qu'un lien avec un nœud voisin est rompu. Un agent *RectifierAnt* est généré pour chaque destination devenue injoignable à cause de cette rupture de lien et envoyé à chaque nœud voisin concerné par cette destination (un voisin $\in$ *Nlist*, voir A.5) afin qu'il puisse à son tour mettre à jour ses informations de routage. Ce processus est répété par chacun de ses voisins et ainsi de suite jusqu'à ce que l'information soit relayée et prise en compte par tous les nœuds concernés.

Dans l'exemple décrit dans la figure A.5, dès que le nœud 1 détecte la rupture du lien avec le nœud 2 grâce aux envois périodiques des *hello* messages, il consulte sa table de routage, sélectionne toutes les destinations concernées par cette rupture (c'est-à-dire, toutes les routes ayant comme nœud suivant le nœud 2), crée un agent *RectifierAnt* et l'envoie aux nœuds voisins contenus dans la colonne *Nlist* de l'entrée sélectionnée. Par exemple pour la destination *dest*3, l'agent *RectifierAnt* est envoyé aux nœuds voisins 5, 3 et 16. Ce processus est réitéré par chacun des nœuds recevant l'agent.

A.3 Simulations

A.3.1 Environnement de simulation

Nous avons testé notre protocole en utilisant le simulateur réseau NS2. Notre protocole a été entièrement développé avec le langage C++ et intégré à la plateforme de simulation NS2 en respectant le même processus de développement que les autres protocoles de routage déjà existants sur cet environnement. Dans chacune des simulations réalisées, les nœuds étaient mobiles avec une vitesse de déplacement maximale de $5m/s$. Le trafic est généré automatiquement et aléatoirement par un script *TCL* mis au point pour ce travail et les scénarios de mobilité sont générés grâce à l'utilitaire *SETDEST* fourni dans l'environnement NS2. Le nombre total de paquets de données varie entre 700 et 1000 paquets par simulation. La taille de chaque paquet est de 512 octets.

Dans ce qui suit, nous allons comparer notre protocole à deux autres protocoles de routage : AODV (un protocole réactif) et DSDV (un protocole proactif). Nous avons choisi un réseau de densité moyenne (d'une superficie de $300m^2$). Nous avons opté pour ce type de réseau, car c'est le plus répandu dans la pratique. Nous avons pris en considération trois critères d'évaluation :

- *pourcentage de perte de paquets* (le nombre de paquets perdus) : il mesure le nombre de paquets qui ne sont pas livrés à leurs destinataires ; Il nous renseigne, en premier lieu,

sur la robustesse de notre protocole et, en second lieu, sur la congestion du réseau.

- *délais de transmission* : cette métrique représente le délai moyen entre le moment d'envoi d'un paquet de données et le moment de sa réception ;
- et enfin, le critère qui nous intéresse le plus, *le nombre de paquets de contrôle générés* : Cette métrique représente le nombre de messages de contrôle (le nombre d'agents dans le cas de notre protocole) émis lors des processus d'établissement et de maintien des routes.

Pour chaque métrique utilisée, nous avons mis en place des jeux de test en faisant varier le nombre de nœuds dans le réseau (le nombre de nœuds varie de vingt à cent) et en faisant varier le nombre total de paquets de données envoyés dans le réseau entre 700 à 1000 (C'est-à-dire, on a varié le nombre total des communications d'une simulation à une autre).

A.3.2 Résultats de simulation

Nombre de paquets de contrôle

Les paquets qui consomment la plus grande partie de la bande passante sont les paquets émis durant les phases de découverte et de maintien des routes. Les figures A.6 et A.7 représentent le nombre de paquets de contrôle émis durant ces deux processus.

Nous constatons une différence considérable entre notre protocole et le protocole AODV. Le nombre de paquets de contrôle varie légèrement et de manière linéaire dans notre protocole, que ce soit en augmentant le nombre de nœuds ou bien en augmentant le trafic réseau, alors qu'il augmente très rapidement dans le cas du protocole AODV. Ceci s'explique par le fait que dans notre cas, le nombre d'agents est géré et contrôlé au niveau de chaque nœud et que ce nombre reste proportionnel au nombre de nœuds dans le réseau et à la fréquence d'envoi d'agents utilisée.

Dans le cas de AODV, le nombre de paquets de contrôle dépend de nombreux facteurs et tout particulièrement du nombre de demandes de routes qui consomment énormément de bande passante étant donné qu'elles nécessitent un certain nombre de diffusions dans tout le réseau. De plus, on pourrait supposer un nombre important de collisions dues essentiellement aux diffusions et à la densité du réseau choisi. La présence de collisions occasionne une augmentation du nombre de tentatives de rémission, ce qui augmente en conséquence le nombre total de paquets dans le réseau.

Une comparaison avec le protocole DSDV montre clairement l'équivalence des résultats entre notre protocole et le protocole DSDV. Ceci met en avant l'efficacité de notre protocole

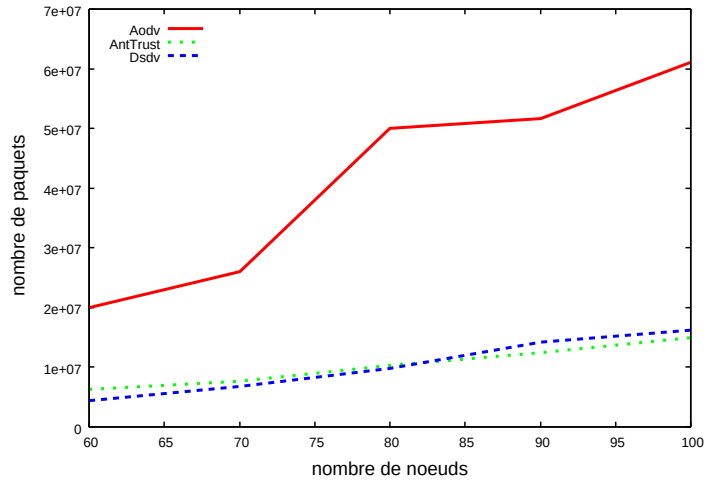

FIGURE A.6 – Coûts en communication suivant le nombre de nœuds dans le réseau.

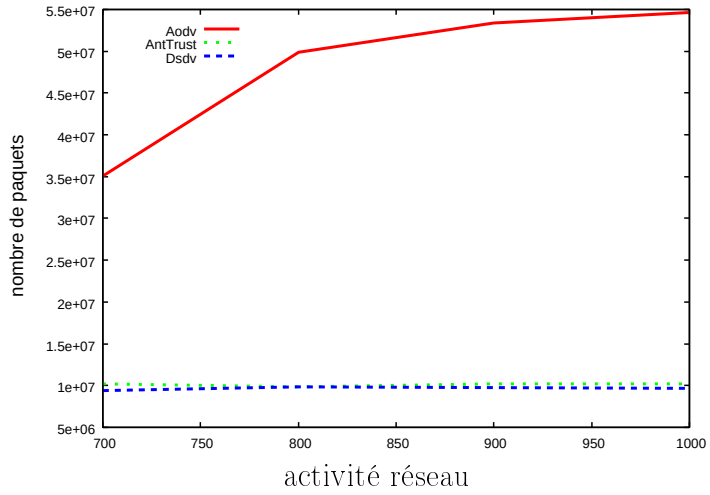

FIGURE A.7 – Coûts en communication suivant le nombre total de paquets de données émis durant la durée de simulation.

dans la mesure où le protocole DSDV est réputé pour être performant dans ce type de réseaux.

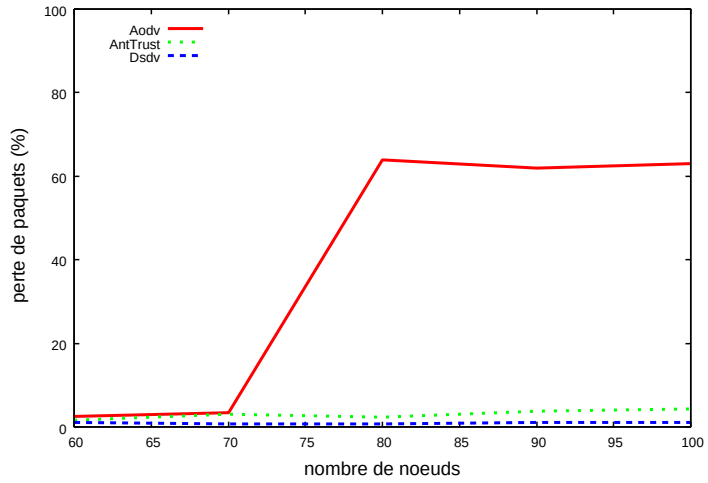

FIGURE A.8 – Pourcentage de perte de paquets suivant le nombre de nœuds dans le réseau.

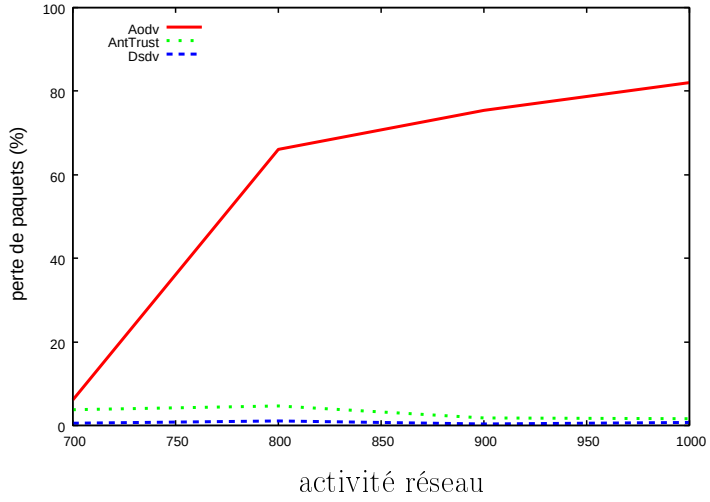

FIGURE A.9 – Pourcentage de perte de paquets suivant le nombre total de paquets de données émis durant la durée de la simulation.

Perte de paquets

Cette métrique nous permet d'évaluer et de comparer la robustesse et l'efficacité des trois protocoles. Il est montré clairement sur les figures A.8 et A.9 que notre protocole et le protocole DSDV accusent beaucoup moins de pertes de paquets que le protocole AODV. On

remarque aussi que le protocole AODV ne s'adapte pas très bien aux réseaux moyennement denses et on le voit très clairement à travers les résultats de simulation présentés sur les deux figures. Le nombre de paquets perdus augmente très rapidement dans des réseaux de plus de 60 nœuds et dans les réseaux à fort trafic. On pourrait expliquer ces résultats par le fait que le protocole AODV utilise un mécanisme de diffusion qui génère un flux de données assez important et ce fait est aggravé par la multiplication de collisions, ce qui surcharge le réseau et laisse un très grand nombre de transmissions non abouties.

Délais de transmission

Cette métrique nous permet de mesurer la rapidité de transmission des trois protocoles de routage étudiés à savoir notre protocole, le protocole DSDV et le protocole AODV. Des temps de transmission trop importants vont influer négativement sur l'utilisation, dans la pratique, de ces protocoles de routage. Les figures A.10 et A.11 nous donnent les durées moyennes de transmission dans chacun des protocoles. D'après les résultats obtenus, nous pouvons affirmer que notre protocole et DSDV (dans le cas de DSDV, on aurait des délais plus importants dans le cas d'un réseau moins dense) engendrent des délais de transmission plus courts que le protocole AODV. La technique de routage utilisée dans notre protocole ainsi que celle utilisée dans le protocole DSDV sont toutes les deux plus efficaces que celle utilisée dans le protocole AODV. En ce qui concerne notre proposition, ceci est dû à deux raisons essentielles :

- le caractère hybride du routage réduit de manière significative les délais de transmission. En effet, grâce à la partie proactive de notre protocole, il n'est pas nécessaire de faire des demandes de routes qui prennent du temps et qui génèrent beaucoup de trafic réseau, car les routes sont déjà disponibles dans les tables de routage ;
- comme notre protocole est multichemin, il permet à chaque nœud d'avoir plusieurs routes (vers une même destination) à chaque fois qu'il veut envoyer un paquet de données et dans lesquelles il peut puiser lors d'une rupture de liens sans relancer une nouvelle procédure de découverte de routes.

A.4 Conclusion

Nous avons voulu apporter ici des solutions pratiques aux problèmes de performance des protocoles de routage destinés aux réseaux ad hoc. Après avoir constaté que la meilleure

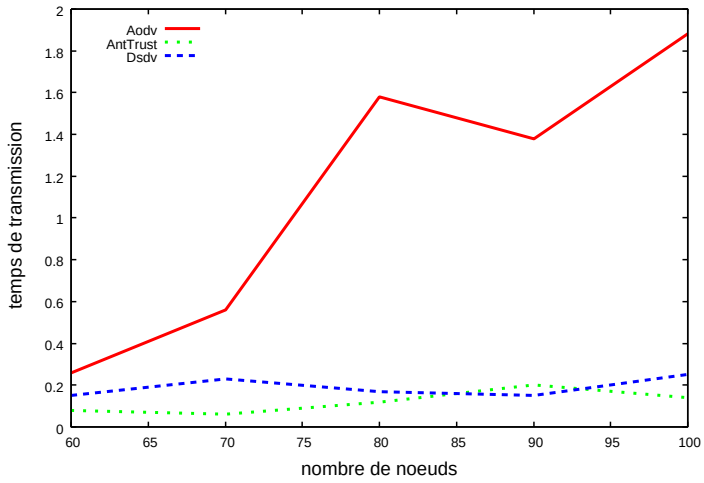

FIGURE A.10 – Délais de transmission suivant le nombre de nœuds.

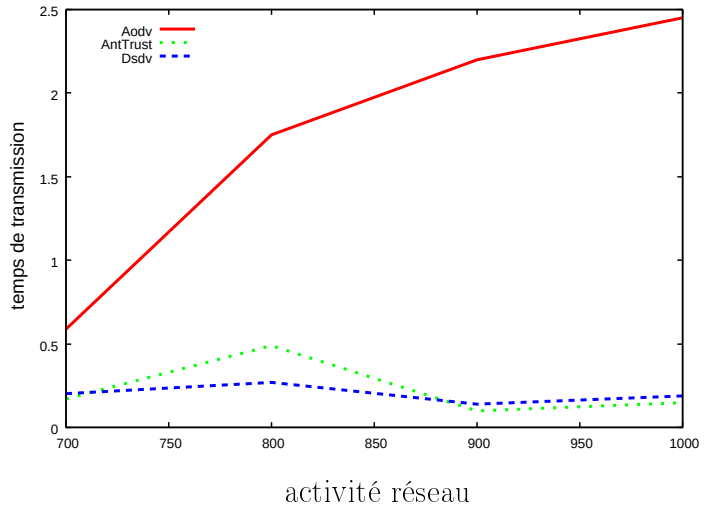

FIGURE A.11 – Délais de transmission suivant le nombre total de paquets de données émis durant la durée de simulation.

manière d'aborder la conception de tels protocoles était d'opter d'une part, pour une approche hybride qui s'appuie sur les avantages des deux méthodes les plus efficaces à savoir l'approche réactive et l'approche proactive et d'autre part, pour une approche basée sur des agents intelligents, nous avons proposé et simulé un protocole de routage hybride basé sur le fonctionnement des colonies de fourmis. Les tests effectués sur une première version de

notre protocole et sur les protocoles AODV et DSDV se sont avérés très encourageants. Sur les trois métriques utilisées dans nos tests, notre proposition se partage les bonnes notes avec le protocole DSDV qui est réputé pour être performant dans le type de réseau que nous avons choisi pour nos tests à savoir un réseau moyennement dense.

Nos objectifs à court terme sont nombreux. Un de ces objectifs est de réaliser d'autres tests sur d'autres environnements et sous d'autres conditions, en utilisant par exemple d'autres métriques et d'autres topologies réseaux. Nous souhaitons aussi apporter dans nos futurs travaux quelques améliorations à cette première version en proposant une implémentation des différentes optimisations que nous avons abordées dans ce chapitre.

Annexe B

Topologies de réseaux de capteurs

Nous avons testé notre protocole de détection active de nœuds répliqués en utilisant une variété de topologies de réseaux de capteurs dont quelques exemples sont exposés ci-dessous.

Figure B.1 – Grand L

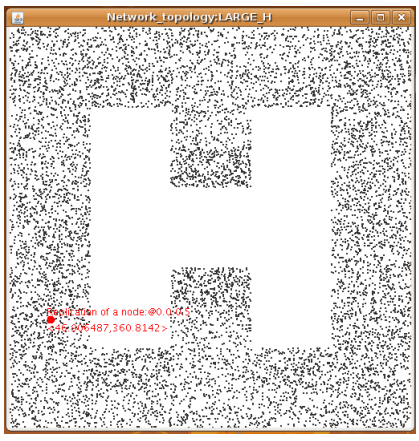

Figure B.2 – Grand H

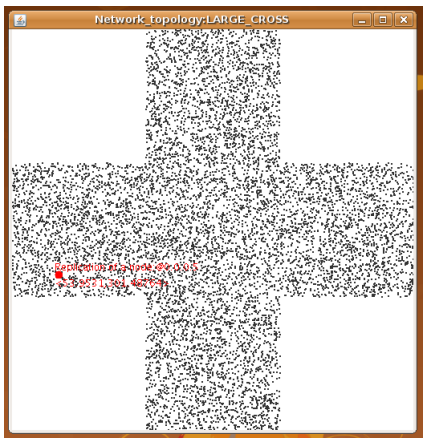

FIGURE B.3 – Grande Croix

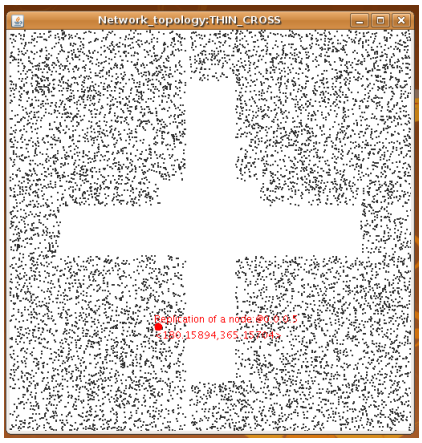

FIGURE B.4 – Croix étroite

FIGURE B.5 – H étroit

FIGURE B.6 – S

Bibliographie

[AAG09] Carlos Aguilar Melchor, Boussad Ait Salem, and Philippe Gaborit. A collusion-resistant distributed scalar product protocol with application to privacy-preserving computation of trust. In *Proceedings of the 2009 Eighth IEEE International Symposium on Network Computing and Applications*, pages 140–147. IEEE Computer Society, 2009.

[AAGT08] Carlos Aguilar Melchor, Boussad Ait Salem, Philippe Gaborit, and Karim Tamine. Anttrust : A novel ant routing protocol for wireless ad-hoc network based on trust between nodes. In *Proceedings of the The Third International Conference on Availability, Reliability and Security, ARES 2008*, pages 1052–1059. IEEE Computer Society, 2008.

[AAGT09] Carlos Aguilar Melchor, Boussad Ait Salem, Philippe Gaborit, and Karim Tamine. Active detection of node replication attacks. *International Journal of Computer Science and Network Security*, 9(2) :13–21, 2009.

[ACL+03] Cedric Adjih, Thomas Clausen, Anis Laouiti, Paul Mühlethaler, and Daniele Raffo. Securing the olsr protocol. In *In 2nd IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2003), Mahdia*, pages 25–27, 2003.

[AD01] Mikhail J. Atallah and Wenliang Du. Secure multi-party computational geometry. pages 165–179. Springer-Verlag, 2001.

[AEDS03] Mikhail Atallah, Hicham Elmongui, Vinayak Deshpande, and Leroy Schwarz. Secure supply-chain protocols. In *2003 IEEE Conference on Electronic Commerce (CEC)*, pages 293–302. IEEE Computer Society, 06 2003.

[ARH97] Alfarez Abdul-Rahman and Stephen Hailes. A distributed trust model. In *NSPW '97 : Proceedings of the 1997 workshop on New security paradigms*, pages 48–60. ACM, 1997.

[ASSC02] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks : a survey. *Computer Networks*, 38(4) :393 – 422, 2002.

[BDT99] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. *From Natural to Artificial Swarm Intelligence*. Oxford University Press, 1999.

[BHvR05] Rimon Barr, Zygmunt J. Haas, and Robbert van Renesse. Jist : an efficient approach to simulation using virtual machines : Research articles. *Softw. Pract. Exper.*, 35(6) :539–576, 2005.

[BK06] E. Barker and J. Kelsey. Recommendation for random number generation using deterministic random bit generators. Technical report, U.S. DoC/National Institute of Standards and Technology, 2006.

[BLB02] Sonja Buchegger and Jean-Yves Le Boudec. Performance analysis of the confidant protocol. In *MobiHoc '02 : Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 226–236. ACM, 2002.

[Bou02] Imed Bouazizi. Ara - the ant-colony based routing algorithm for manets. In *ICPPW '02 : Proceedings of the 2002 International Conference on Parallel Processing Workshops*, page 79. IEEE Computer Society, 2002.

[CBH03] Srdjan Capkun, Levente Buttyan, and Jean-Pierre Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, pages 52–64, 2003.

[CBS05] Gilbert Chen, Joel W. Branch, and Boleslaw K. Szymanski. Local leader election, signal strength aware flooding, and routeless routing. In *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Workshop 12 - Volume 13*, IPDPS '05, pages 244.1–. IEEE Computer Society, 2005.

[CCBNR07] F. Cuppens, N. Cuppens-Boulahia, S. Nuon, and T. Ramard. Property based intrusion detection to secure olsr. In *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Communications (ICWMC'07)*, pages 52–60. IEEE Computer Society, 2007.

[CDPMM07] Mauro Conti, Roberto Di Pietro, Luigi Vincenzo Mancini, and Alessandro Mei. A randomized, efficient, and distributed protocol for the detection of node replication attacks in wireless sensor networks. In *Proceedings of the 8th ACM*

international symposium on Mobile ad hoc networking and computing, MobiHoc '07, pages 80–89. ACM, 2007.

[Cha88] David Chaum. The dining cryptographers problem : Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1 :65–75, 1988.

[CJ07] Hung-Chi Chu and Rong-Hong Jan. A gps-less, outdoor, self-positioning method for wireless sensor networks. *Ad Hoc Netw.*, 5(5) :547–557, 2007.

[CJI02] Bled Electronic Commerce, Audun Jøsang, and Roslan Ismail. The beta reputation system. In *In Proceedings of the 15th Bled Electronic Commerce Conference*. Citeseer, 2002.

[CKV+02] Chris Clifton, Murat Kantarcioglu, Jaideep Vaidya, Xiaodong Lin, and Michael Y. Zhu. Tools for privacy preserving distributed data mining. *SIGKDD Explor. Newsl.*, 4(2) :28–34, 2002.

[CLR90] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press, 1990.

[CPS03] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, SP '03, pages 197–. IEEE Computer Society, 2003.

[DBDAJ01] Johnson David B, Maltz David A, and Broch Josh. Dsr : The dynamic source routing protocol for multi-hop wireless ad hoc networks. In *In Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5*, pages 139–172. Addison-Wesley, 2001.

[DDC99] Marco Dorigo and Gianni Di Caro. The ant colony optimization meta-heuristic. pages 11–32, 1999.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, STOC '91, pages 542–552. ACM, 1991.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22 :644–654, 1976.

[DJ01] Ivan Damgard and Mats Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *PKC '01 :*

Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography, pages 119–136. Springer-Verlag, 2001.

[Dou02] John R. Douceur. The sybil attack. In *IPTPS '01 : Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260. Springer-Verlag, 2002.

[DPG01] Lance Doherty, Kristofer S. J. Pister, and Laurent El Ghaoui. Convex optimization methods for sensor node position estimation. In *INFOCOM*, pages 1655–1663, 2001.

[EG85] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18. Springer-Verlag New York, Inc., 1985.

[EGB02] Laurent Eschenauer, Virgil D. Gligor, and John Baras. On trust establishment in mobile ad-hoc networks. In *In Proceedings of the Security Protocols Workshop*, pages 47–66. Springer-Verlag, 2002.

[FC01] Rino Falcone and Cristiano Castelfranchi. Social trust : a cognitive approach. pages 55–90, 2001.

[Fei99] Uriel Feige. Noncryptographic selection protocols. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, FOCS '99, pages 142–. IEEE Computer Society, 1999.

[Gam88] Diego Gambetta. *Can We Trust Trust ?* Basil Blackwell, 1988.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178. ACM, 2009.

[GFLM05] Di Caro Gianni, Ducatelle Frederick, and Gambardella Luca Maria. Anthocnet : An adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, 16 :443–455, 2005.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4) :792–807, 1986.

[GLLM04] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On private scalar product computation for privacy-preserving data mining. In Choonsik Park and Seongtaek Chee, editors, *Information Security and Cryptology - ICISC*

2004, 7th International Conference, Seoul, Korea, December 2-3, 2004, Revised Selected Papers, volume 3506, pages 104–120. Springer, 2004.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2) :270–299, 1984.

[Gol04] Oded Goldreich. *Foundations of Cryptography : Volume 2, Basic Applications*. Cambridge University Press, 2004.

[Gol05] Oded Goldreich. Foundations of cryptography : a primer. *Found. Trends Theor. Comput. Sci.*, 1(1) :1–116, 2005.

[Haa97] Zygmunt J. Haas. A new routing protocol for the reconfigurable wireless networks. In *Proceedings of 6th IEEE International Conference on Universal Personal Communications, IEEE ICUPC'97, October 12-16, 1997, San Diego, California, USA*, volume 2, pages 562–566. IEEE, IEEE, 1997.

[HBC01] Jean-Pierre Hubaux, Levente Buttyán, and Srdan Capkun. The quest for security in mobile ad hoc networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '01, pages 146–155. ACM, 2001.

[HBH+05] Carl Hartung, James Balasalle, Richard Han, Carl Hartung, James Balasalle, and Richard Han. Node compromise in sensor networks : The need for secure systems. Technical report, 2005.

[HE04] Lingxuan Hu and David Evans. Using directional antennas to prevent wormhole attacks. In *NDSS*. The Internet Society, 2004.

[Hed88] C. L. Hedrick. Routing information protocol. 1988.

[HJP02] Yih-Chun Hu, David B. Johnson, and Adrian Perrig. Sead : Secure efficient distance vector routing for mobile wireless ad hoc networks. *Mobile Computing Systems and Applications, IEEE Workshop on*, 0 :3, 2002.

[HMM+00] Amir Herzberg, Yosi Mass, Joris Michaeli, Yiftach Ravid, and Dalit Naor. Access control meets public key infrastructure, or : Assigning roles to strangers. In *SP '00 : Proceedings of the 2000 IEEE Symposium on Security and Privacy*, page 2. IEEE Computer Society, 2000.

[HP01] Zygmunt J. Haas and Marc R. Pearlman. Zrp : a hybrid framework for routing in ad hoc networks. pages 221–253, 2001.

[HPJ03] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Packet leashes : A defense against wormhole attacks in wireless networks. pages 1976–1986 vol.3. IEEE Computer Society, apr 2003.

[HPJ05] Yih-Chun Hu, Adrian Perrig, and David B. Johnson. Ariadne : a secure on-demand routing protocol for ad hoc networks. *Wirel. Netw.*, 11(1-2) :21–38, 2005.

[HTR+04] Andreas Hafslund, Andreas Tønnesen, Roar Bjørgum Rotvik, Jon Andersson, and Øivind Kure. Secure extensions to the olsr protocol. In *In OLSR Interop Workshop*, 2004.

[HWS09] Chung-Wei Hang, Yonghong Wang, and Munindar P. Singh. Operators for propagating trust and their evaluation in social networks. In *AAMAS '09 : Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 1025–1032. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[JIB07] Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decis. Support Syst.*, 43(2) :618–644, 2007.

[JP05] Audun Jøsang and Simon Pope. Semantic constraints for trust transitivity. In *APCCM '05 : Proceedings of the 2nd Asia-Pacific conference on Conceptual modelling*, pages 59–68. Australian Computer Society, Inc., 2005.

[KK00] Brad Karp and H. T. Kung. Gpsr : greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, pages 243–254. ACM, 2000.

[KRD06] Sunil Kumar, Vineet S. Raghavan, and Jing Deng. Medium access control protocols for ad hoc wireless networks : A survey. *Ad Hoc Netw.*, 4(3) :326–358, 2006.

[KZL+01] Jiejun Kong, Petros Zerfos, Haiyun Luo, Songwu Lu, and Lixia Zhang. Providing robust and ubiquitous security support for mobile ad-hoc networks. In *ICNP '01 : Proceedings of the Ninth International Conference on Network Protocols*, pages 251–260. IEEE Computer Society, 2001.

[LKZ+04] Haiyun Luo, Jiejun Kong, Petros Zerfos, Songwu Lu, and Lixia Zhang. Ursa : Ubiquitous and robust access control for mobile ad-hoc networks. *IEEE/ACM Transactions on Networking*, 12 :1049–1063, 2004.

[LL99] Chunhung Richard Lin and Jain-Shing Liu. Qos routing in ad hoc wireless networks. *Selected Areas in Communications, IEEE Journal on*, 17(8) :1426 –1438, aug 1999.

[LN08] An Liu and Peng Ning. Tinyecc : A configurable library for elliptic curve cryptography in wireless sensor networks. In *IPSN '08 : Proceedings of the 7th international conference on Information processing in sensor networks*, pages 245–256. IEEE Computer Society, 2008.

[LP06] Jeong-Mi Lim and Chang-Seop Park. Session key agreement protocol for end-to-end security in manet. In Marina Gavrilova, Osvaldo Gervasi, Vipin Kumar, C. Tan, David Taniar, Antonio Laganá, Youngsong Mun, and Hyunseung Choo, editors, *Computational Science and Its Applications - ICCSA 2006*, volume 3983 of *Lecture Notes in Computer Science*, pages 679–686. Springer Berlin / Heidelberg, 2006.

[MC96] D. Harrison Mcknight and Norman L. Chervany. The meanings of trust, 1996.

[MGLB00] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *MobiCom '00 : Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 255–265. ACM, 2000.

[mic04] MICAz - wireless measurement system. Technical report, 2004.

[MM02] Pietro Michiardi and Refik Molva. Core : a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security*, pages 107–121. Kluwer, B.V., 2002.

[MMA+01] Lik Mui, Mojdeh Mohtashemi, Cheewee Ang, Peter Szolovits, and Ari Halberstadt. Ratings in distributed systems : A bayesian approach, 2001.

[MMH02] L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation. pages 2431 – 2439. IEEE Computer Society, jan. 2002.

[MvOV01] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 2001.

[NS98] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *CCS '98 : Proceedings of the 5th ACM conference on Computer and communications security*, pages 59–66. ACM, 1998.

[NS03] James Newsome and Dawn Song. Gem : Graph embedding for routing and data-centric storage in sensor networks without geographic information. In *SenSys '03 : Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 76–88. ACM, 2003.

[NSSP04] James Newsome, Elaine Shi, Dawn Song, and Adrian Perrig. The sybil attack in sensor networks : analysis & defenses. In *IPSN '04 : Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 259–268. ACM, 2004.

[OLS03] Optimized link state routing protocol (olsr), 2003.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. pages 223–238. 1999.

[PB94] Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. In *Proceedings of the conference on Communications architectures, protocols and applications*, pages 234–244, 1994.

[PBRD03] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing, 2003.

[PH02] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure routing for mobile ad hoc networks. In *SCS COMMUNICATION NETWORKS AND DISTRIBUTED SYSTEMS MODELING AND SIMULATION CONFERENCE (CNDS 2002)*, pages 193–204, 2002.

[PH09] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization : Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2009.

[PPG05] Bryan Parno, Adrian Perrig, and Virgil Gligor. Distributed detection of node replication attacks in sensor networks. In *SP '05 : Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 49–63. IEEE Computer Society, 2005.

[PRT04] Elan Pavlov, Jeffrey S. Rosenschein, and Zvi Topol. Supporting privacy in decentralized additive reputation. In *Trust Management*, volume 2995, pages 108–119. Springer Berlin / Heidelberg, 2004.

[RACM04] Daniele Raffo, Cédric Adjih, Thomas Clausen, and Paul Mühlethaler. An advanced signature system for olsr. In *SASN '04 : Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 10–16. ACM, 2004.

[RAD78] R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. pages 169–177. Academic Press, 1978.

[RL95] Y. Rekhter and T. Li. A border gateway protocol 4 (bgp-4), 1995.

[RS92] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '91, pages 433–444. Springer-Verlag, 1992.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2) :120–126, 1978.

[RZ02] Paul Resnick and Richard Zeckhauser. Trust among strangers in Internet transactions : Empirical analysis of eBay's reputation system. In Michael R. Baye, editor, *The Economics of the Internet and E-Commerce*, volume 11 of *Advances in Applied Microeconomics*, pages 127–157. Elsevier Science, 2002.

[Sak89] Michael E. Saks. A robust noncryptographic protocol for collective coin flipping. *SIAM J. Discrete Math.*, 2(2) :240–244, 1989.

[SDB03] Brian Shand, Nathan Dimmock, and Jean Bacon. Trust for ubiquitous, transparent collaboration. *Pervasive Computing and Communications, IEEE International Conference on*, 0 :153, 2003.

[sec02] A secure routing protocol for ad hoc networks. In *Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 78–89. IEEE Computer Society, 2002.

[Sha49] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, pages 656–715, 10 1949.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11) :612–613, 1979.

[SPvDK04] Arvind Seshadri, Adrian Perrig, Leendert van Doorn, and Pradeep Khosla. Swatt : Software-based attestation for embedded devices. volume 0, page 272. IEEE Computer Society, 2004.

[Sti05] Douglas R. Stinson. *Cryptography : Theory and Practice*. Chapman & Hall/CRC, 2005.

[TB04] George Theodorakopoulos and John S. Baras. Trust evaluation in ad-hoc networks. In *WiSe '04 : Proceedings of the 3rd ACM workshop on Wireless security*, pages 1–10. ACM, 2004.

[THVW05] Huu Tran, Michael Hitchens, Vijay Varadharajan, and Paul Watters. A trust based access control framework for p2p file-sharing systems. In *HICSS '05 : Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, page 302.3. IEEE Computer Society, 2005.

[Wai90] Michael Waidner. Unconditional sender and recipient untraceability in spite of active attacks. In *Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*, pages 302–319. Springer-Verlag New York, Inc., 1990.

[WGS+09] Xun Wang, Wenjun Gu, Kurt Schosek, Sriram Chellappan, and Dong Xuan. Sensor network configuration under physical attacks. *Int. J. Ad Hoc Ubiquitous Comput.*, 4 :174–182, April 2009.

[WGYX05] Xun Wang, Wenjun Gu, Wei Yu, and Dong Xuan. Search-based physical attacks in sensor networks : Modeling and defense. Technical report, in Proc. of International Conference on Computer Communications and Networks (ICCCN, 2005.

[WJI04] Andrew Whitby, Audun Jøsang, and Jadwiga Indulska. Filtering out unfair ratings in bayesian reputation systems. In *Workshop on Trust in Agent Societies*, number 2, pages 106–117. Citeseer, 2004.

[WKvO04] Tao Wan, Evangelos Kranakis, and Paul van Oorschot. Securing the destination-sequenced distance vector routing protocol (s-dsdv). In Javier Lopez, Sihan Qing, and Eiji Okamoto, editors, *Information and Communications Security*, volume 3269, pages 311–318. Springer Berlin / Heidelberg, 2004.

[WLMG05] M. Wang, L. Lamont, P Mason, and M. Gorlatova. An effective intrusion detection approach for olsr manet protocol. In *Proceedings of the First Workshop on Secure Network Protocols (NPSec), Boston, Massachusetts, États-Unis.*, jul 2005.

[Yao82] A. Yao. Protocols for secure computations. In *the twenty-third annual IEEE Symposium on Foundations of Computer Science*, pages 160–164. IEEE Computer Society, 1982.

[YTP07] Danfeng Yao, Roberto Tamassia, and Seth Proctor. Private distributed scalar product protocol with application to privacy-preserving computation of trust. In *IFIPTM 2007 – Joint iTrust and PST Conferences on Privacy, Trust Management and Security*. Springer Boston, July 2007.

[Zap02] Manel Guerrero Zapata. Secure ad hoc on-demand distance vector routing, June 2002.

[ZAS+07] Bo Zhu, Venkata Gopala Krishna Addada, Sanjeev Setia, Sushil Jajodia, and Sankardas Roy. Efficient distributed detection of node replication attacks in sensor networks. *Computer Security Applications Conference, Annual*, 0 :257–267, 2007.

[ZH99] Lidong Zhou and Zygmunt Haas. Securing ad hoc networks. *IEEE Network Magazine*, 13 :24–30, 1999.

[Zim95] Philip R. Zimmermann. *The official PGP user's guide*. MIT Press, 1995.